НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп‘ютерних систем

«До захисту допущено»

Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ

“___” червня 2020 р.

Дипломний проект

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Спеціалізовані комп’ютерні системи»

зі спеціальності **123 «Комп'ютерна інженерія»**

на тему: Вебдодаток для обліку коштів та планування витрат

Виконав: студент IV курсу, групи КВ-61
(шифр групи)

Хлоп’ячий Павло Андрійович_____
(прізвище, ім’я, по батькові) (підпис)

Керівник доц. каф. СПіСКС, к.т.н., доцент Сапсай Т.Г. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з нормоконтролю, доц.каф.СПіСКС, к.т.н. Клятченко Я.М.

(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	ДП 045440.004 ПЗ	Пояснювальна записка	50	
3	А4	ДП 045440.005 Д1	Робота серверу під час API виклику. Схема алгоритму	1	
4	А4	ДП 045440.006 Д2	Принцип взаємодії модулів. Структурна схема.	1	
5	А4	ДП 045440.007 Д3	Сутності бази даних. Структурна схема.	1	
6	А4	ДП 045440.008 Д4	Вебдодаток для обліку коштів та контролювання витрат. Схема алгоритму	1	

				ДП 045440.000		
	ПІБ	Підп.	Дата			
Розробн.	Хлоп’ячий П.А.			Відомість дипломного проєкту	Лист	Листів
Керівн.	Сапсай Т.Г.				1	1
Консульт.					КПІ ім. Ігоря Сікорського Каф. СПіСКС Гр. КВ-61	
Н/контр.	Клятченко Я.М.					
Зав.каф.	Романкевич В.О.					

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 123 «Комп'ютерна інженерія»

Освітньо-професійна програма «Спеціалізовані комп'ютерні системи»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ
(підпис) (ініціали, прізвище)
«__»_____20___р.

ЗАВДАННЯ
на дипломний проєкт студента

Хлоп'ячого Павла Андрійовича

1. Тема проєкту «Вебдодаток для обліку коштів та планування витрат», керівник проєкту доц. каф. СПіСКС, к.т.н., доцент Сапсай Т.Г.,

затверджені наказом по університету від «__»_____20__р. №_____

2. Термін подання студентом проєкту 28.05.2020

3. Вихідні дані до проєкту Назва. Вебдодаток для обліку коштів та планування витрат

4. Зміст пояснювальної записки
   1. Аналіз існуючих вебдодатків для обліку коштів та контролювання витрат
   2. Аналіз обраних технологій для розробки дипломного проєкту
   3. Вебдодаток для обліку коштів та планування витрат

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо) презентація; блок-схема роботи серверу під час

API виклику; структурна схема взаємодії модулів дворівневої системи запитів; структурна схема сутностей бази даних; блок-схема процесу створення цілі.

6. Консультанти розділів проєкту*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
нормоконтроль	Ярослав КЛЯТЧЕНКО		

7. Дата видачі завдання 17.09.2019

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Видача завдання на дипломне проєктування	30.10.2019	
1.	Вивчення літератури за тематикою проєкту	09.11.2019	
2.	Розроблення та узгодження технічного завдання	18.11.2019	
3.	Аналіз існуючих рішень	12.12.2019	
4.	Підготовка матеріалів першого розділу дипломного проєкту	28.02.2020	
5.	Підготовка матеріалів другого та третього розділів дипломного проєкту	15.04.2020	
6.	Підготовка графічної частини дипломного проєкту	19.05.2020	
7.	Оформлення документації дипломного проєкту	20.05.2020	
8.	Попередній огляд матеріалів проєкту на кафедрі	22.05.2020	

Студент _____ Павло ХЛОП’ЯЧИЙ
Керівник проєкту _____ Тетяна САПСАЙ

_____

АНОТАЦІЯ

Дипломний проект присвячений розробці вебдодатку для обліку коштів та планування витрат, котрий включає в себе пояснювальну записку (50 стор., 31 рис.).

При розробці даного проєкту було досягнено наступні цілі:

- проведено аналіз існуючих додатків;
- проаналізовано методи та технології для розробки вебдодатків;
- розроблено архітектуру вебдодатку.

Під час розробки була використана мова програмування JavaScript із застосуванням бібліотек React та Redux для клієнтського інтерфейсу, а також фреймворку Express для середовища виконання NodeJS для серверної частини програми, Visual Studio Code було використано у якості середовища розробки.

У процесі розробки було розроблено вебдодаткок для обліку коштів та планування витрат, який задовольняє такі умови:

- використання з будь-якого самостійного пристрою, який містить браузер;
- реєстрації та авторизації користувача;
- додавання прибутків та витрат;
- отримання графічного аналізу даних.

Ключові слова: вебдодаток, база даних, JavaScript, React, Redux, NodeJS, Express, Internet, API, MongoDB.

ABSTRACT

The diploma project is devoted to the development of a web application for accounting and cost planning, which includes an explanatory note (50 pages, 31 figures).

The following goals were achieved during the development of this project:

- the analysis of existing applications is carried out;
- methods and technologies for web application development are analyzed;
- web application architecture developed.

During the development, the JavaScript programming language was used using the React and Redux libraries for the client interface, as well as the Express framework for the NodeJS runtime for the server part of the program, and Visual Studio Code was used as the development environment.

In the process of development, a web application was developed for cost accounting and cost planning, which meets the following conditions:

- use from any device that contains a browser;
- user registration and authorization;
- adding profits and expenses;
- obtaining graphical data analysis.

Keywords: web application, database, JavaScript, React, Redux, NodeJS, Express, Internet, API, MongoDB.

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
			Документація загальна			
			Новорозроблена			
	А4	ІАЛЦ.045440.002 ТЗ	Вебдодаток для обліку	3		
			коштів та контролювання			
			витрат.			
			Технічне завдання			
	А4	ІАЛЦ.045440.003 ТП	Вебдодаток для обліку	2		
			коштів та контролювання			
			витрат.			
			Відомість технічного			
			проекту			
	А4	ІАЛЦ.045440.004 ПЗ	Вебдодаток для обліку	50		
			коштів та контролювання			
			витрат.			
			Пояснювальна записка			
	А4	ІАЛЦ.045440.005 Д1	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Робота серверу під час API			
			виклику.			
			Схема алгоритму			

Змін.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.045440.001 ОА		
Розробив		Хлоп'ячий П.А.			*Вебдодаток для обліку коштів та контролювання витрат* *Опис альбому*	Літ.	Аркуш 1 / Аркушів 2
Перевірив		Сапсай Т. Г.					
Н. контроль		Клятченко Я.М.				КПІ ім. Ігоря Сікорського ФПМ КВ-61	
Затвердив		Романкевич В.О.					

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045440.006 Д2	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Принцип взаємодії модулів			
			дворівневої системи			
			запитів.			
			Схема структурна			
	А4	ІАЛЦ.045440.007 Д3	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Сутності бази даних.			
			Схема структурна			
	А4	ІАЛЦ.045440.008 Д4	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Процес створення цілі.			
			Схема алгоритму			
		Диск CD-ROM	Текст пояснювальної	1		
			записки.			
			Графічний матеріал			

Зміст

1. НАЙМЕНУВАННЯ І ГАЛУЗЬ ЗАСТОСУВАННЯ 2
2. ПІДСТАВА ДЛЯ РОЗРОБКИ 2
3. ЦІЛЬ ТА ПРИЗНАЧЕННЯ ПРОЄКТУ 2
4. ДЖЕРЕЛА РОЗРОБКИ 2
5. ТЕХНІЧНІ ВИМОГИ 3
5.1. Вимоги до системи, що розробляється 3
5.2. Вимоги до програмного забезпечення 3
5.3. Вимоги для апаратного забезпечення 3
6. ЕТАПИ РОЗРОБКИ 4

1. НАЙМЕНУВАННЯ І ГАЛУЗЬ ЗАСТОСУВАННЯ

Назва розробки: «Вебдодаток для обліку коштів та планування витрат».
Галузь застосування: інформаційні технології.

2. ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки даного проєкту є завдання на виконання роботи для отримання бакалаврського рівня освіти, яке затверджене кафедрою системного програмування і спеціалізованих комп'ютерних систем НТУУ «Київський Політехнічний Інститут імені Ігоря Сікорського»

3. МЕТА ТА ПРИЗНАЧЕННЯ ПРОЄКТУ

Метою даного проєкту є створення вебдодатку для обліку коштів та планування витрат.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом інформації для даного проєкту є науково-технічна література, технічна документація та електронні статті у мережі Інтернет.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до програмного продукту, що розробляється:

- сумісність з будь-якою операційною системою, яка має в наявності браузер для перегляду вебсторінок;
- можливість реєструватися на платформі;
- додавання надходжень до бюджету;
- додавання витрат;

- створення цілей;
- отримання аналізу інформації.

5.2. Вимоги до програмного забезпечення:

- операційна система: Windows, Linux, Mac OS;
- наявність браузеру для перегляду вебсторінок;
- наявність доступу до мережі Інтернет.

5.3. Вимоги до апаратного забезпечення:

- 4-хядерний процесор Intel;
- оперативна пам'ять: 8 ГБ;
- наявність доступу до мережі Інтернет.

6. ЕТАПИ РОЗРОБКИ

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Видача завдання на дипломне проєктування	30.10.2019	
2.	Вивчення літератури за тематикою проєкту	09.11.2019	
3.	Розроблення та узгодження технічного завдання	18.11.2019	
4.	Аналіз існуючих рішень	12.12.2019	
5.	Підготовка матеріалів першого розділу дипломного проекту	28.02.2020	
6.	Підготовка матеріалів другого та третього розділів дипломного проєкту	15.04.2020	
7.	Підготовка графічної частини дипломного проєкту	19.05.2020	
8.	Оформлення документації дипломного проєкту	20.05.2020	

9. Попередній огляд матеріалів проєкту на кафедрі — 22.05.2020

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045440.004 ПЗ	Вебдодаток для обліку	50		
			коштів та контролювання			
			витрат.			
			Пояснювальна записка			
	А4	ІАЛЦ.045440.005 Д1	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Робота серверу під час API			
			виклику.			
			Схема алгоритму			
	А4	ІАЛЦ.045440.006 Д2	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Принцип взаємодії модулів			
			дворівневої системи			
			запитів.			
			Схема структурна			
	А4	ІАЛЦ.045440.007 Д3	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Сутності бази даних.			
			Схема структурна			

Змін.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ. 045440.003 ТП		
Розробив		Хлоп'ячий П.А.			Вебдодаток для обліку коштів та контролювання витрат Відомість технічного проекту	Літ.	Аркуш 1 / Аркушів 2
Перевірив		Сапсай Т. Г.					
Консульт.						КПІ ім. Ігоря Сікорського ФПМ КВ-61	
Н. контроль		Клятченко Я.М.					
Зав. каф.		Романкевич В.О.					

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045440.008 Д4	Вебдодаток для обліку	1		
			коштів та контролювання			
			витрат.			
			Процес створення цілі.			
			Схема алгоритму			
		Диск CD-ROM	Текст ПЗ. Тексти програм.	1		
			Графічний матеріал.			

Пояснювальна записка
до дипломного проєкту

на тему: Вебдодаток для обліку коштів та планування витрат

Київ – 2020 року

ЗМІСТ

Перелік скорочень, умовних позначень, термінів 3
ВСТУП 5
1. АНАЛІЗ ІСНУЮЧИХ ВЕБДОДАТКІВ ДЛЯ ОБЛІКУ КОШТІВ ТА ПЛАНУВАННЯ ВИТРАТ 6
1.1. Аналіз існуючих рішень 6
1.2. Завдання на дипломний проєкт 8
2. АНАЛІЗ ОБРАНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ДИПЛОМНОГО ПРОЄКТУ 9
2.1. Мова програмування JavaScript 9
2.2. Інструмент створення інтерфейсів користувача React 11
2.3. Менеджер станів Redux 14
2.4. Використання середовища NodeJS для реалізації серверної частини програми 16
2.5. Вебфреймворк Express 16
2.6. База даних MongoDB 17
2.7. Пакувальник модулів Webpack 19
2.8. Предпроцесор CSS 22
3. ВЕБДОДАТОК ДЛЯ ОБЛІКУ КОШТІВ ТА ПЛАНУВАННЯ ВИТРАТ 25
3.1. Вебдодаток. Його переваги, складові та види 25
3.2. Розробка дизайну та архітектури 30
3.3. Колекції серверної частини програми 31
3.4. Реалізація аутентифікації для користувача 34

3.5. Розробка клієнтської частини додатку.....38
ВИСНОВКИ 48
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....49
Додатки
Додаток 1. Копії графічного матеріалу
ІАЛЦ.045440.005 Д1. Робота серверу під час API виклику. Схема алгоритму.
ІАЛЦ.045440.006 Д2. Принцип взаємодії модулів дворівневої системи запитів. Схема структурна.
ІАЛЦ.045440.007 Д3. Сутності бази даних. Схема структурна.
ІАЛЦ.045440.008 Д4. Процес створення цілі. Схема алгоритму.
Додаток 2. Фрагменти коду
Додаток 3. Презентація

Перелік умовних позначень, термінів

DOM (Document object model) – представлення документу HTML у вигляді дерева тегів;

XML – Extensible Markup Language – розширювана мова розмітки

HTML – HyperText Markup Language – мова розмітки гіпертекстових документів;

XHTML – Extensible HyperText Markup Language – розширювана мова розмітки гіпертексту;

CSS – Cascading Style Sheets – мова описання стилів вебсторінки, що виконана за допомогою HTML;

HTTP – HyperText Transfer Protocol – протокол передачі даних;

СУБД – система управління базами даних

БД – база даних;

JWT – JSON Web Token – стандарт передачі інформації;

API – Application Programming Interface – прикладний програмний інтерфейс;

LS – local Storage – місце зберігання інформації у браузері;

JS – JavaScript – мова програмування

TS – TypeScript – мова програмування основана на JavaScript

ПЗ – програмне забезпечення

ООП – Об'єктно-орієнтоване програмування.

URL – Uniform Resource Locator – стандартизована адреса певного ресурсу;

WWW – World Wide Web – всесвітня мережа;

IDE – Integrated development environment – стандартне інтегроване середовище розробки;

АОП – аспектно-орієнтоване програмування;

AJAX – асинхронний JavaScript сценарій;

OS – Operating system – операційна система;

PHP – Hypertext Preprocessor – мова програмування;

UI – User interface – інтерфейс користувача;

ВСТУП

За даними 2019 року, більше 53% населення Землі, а саме 4,1 мільярдів людей мають доступ до мережі Інтернет. Саме цей факт є джерелом зростання кількості вебресурсів у різноманітних сферах людської життєдіяльності. Вже сьогодні існує багато вебдодатків, робота яких спрямована на покращення життя людини, а також, щоб зробити його більш комфортним. Одним із прикладів такого веб-ресурсу є додаток для контролювання витрат. На сьогоднішній день це є актуальною проблемою для людей різних вікових категорій. Наприклад, люди, котрі тільки почали самостійно заробляти собі на життя та не мають достатньо досвіду для прогнозування та розрахування своїх коштів на певний період. Або для людей середнього віку, котрі вже можуть розрахувати витрати на певний період, але ставлять перед собою вже більш вагомі цілі, то додаток зможе допомогти виділяти відповідну кількість грошей, що не нашкодить бюджету в інших аспектах. Та для людей пенсійного віку, котрі мають обмежені кошти та мусять розраховувати витрати максимально точно.

Завдання дипломного проєкту є розробка веб-орієнтованого додатку для контролювання витрат. Особливістю даного проєкту є розробка саме веб-орієнтовного додатку. На відміну від аналогів вебдодаток не потребуватиме окремої пам’яті для збереження, бо на сучасних телефонах або планшетах завжди є додаток для користування інтернетом, що повністю задовольняє потреби, які необхідні додатку отриманого після завершенню цього проєкту. Додаток повинен забезпечувати процеси авторизації та реєстрації користувачів, заповнення та редагування профілю, можливості для введення та контролю витрат, розрахунку витрат для придбання речей та системи аналізу витрат.

1. АНАЛІЗ ІСНУЮЧИХ ВЕБДОДАТКІВ ДЛЯ ОБЛІКУ КОШТІВ

1.1. Аналіз існуючих рішень

Проаналізуємо існуючі аналоги для мобільних пристроїв.

Станом на 2018 рік сайт Gazeta.ua опублікувала список з відомих додатків для системи Android та IOS, які допомагають контролювати власні кошти та витрати.

Мобільний додаток Wallet

Цей додаток має сучасний дизайн. Для використання необхідно зареєструватися та авторизуватися в системі. Після реєстрації в системі є можливість обрати валюту та задати поточний бюджет. Також передбачено додавання декількох людей до групи за для контролювання сімейного бюджету.

Перевагами даного ресурсу є:

- доступність для користувачів з будь-якою операційною системою;
- можливість створення списків бажань та цілей при розподілі коштів;
- створення та відображення статистичних результатів;
- можливість синхронізації з банківською карткою;

Недоліки додатку Wallet:

- не має змоги змінювати мову інтерфейсу;
- для отримання детальних діаграм та графіків витрат, а також створення певної кількості груп і банківських карт необхідно придбати преміум версію додатку.

Мобільний додаток Monefy

Простий додаток для контролювання фінансів - на головному екрані є дві кнопки: надходження і витрати. Всі грошові операції сортуються за категоріями і виносяться на велику кругову діаграму на стартовому екрані. Категорії можна додавати самостійно. Додаток дозволяє планувати місячні витрати і не виходити за їх межі, статистика буде формуватися відносно заданого бюджету, а не всіх надходжень.

Переваги додатку Monefy:

- можливість вести підрахунок витрат окремо для готівки та банківських карт;
- велика кількість категорій: дім, спорт, транспорт, продукти, одяг, медикаменти тощо;
- отримання даних про витрати за конкретний період;
- розрахування діаграми витрат одразу після їх введення;
- додавання коментарів до витрат;

Недоліки:

- відсутність реєстрації користувача;
- встановлення паролю можливе лише для платної версії додатку;
- використання декількох валют також потребує придбання розширеної версії додатку.

1.2. Завдання на дипломний проєкт

У процесі розробки дипломного проєкту запропоновано реалізацію наступних задач, які забезпечують:

- реєстрацію користувача;
- можливість введення надходжень та витрат;
- класифікацію витрат за категоріями;
- можливість введення циклічних витрат;
- визначення цілей для придбання на певний період.

2. АНАЛІЗ ОБРАНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ДИПЛОМНОГО ПРОЄКТУ

Для виконання веборієнтованих додатків існує велика кількість засобів. Найпоширенішими є: Java, Ruby, .NET, Python для серверної частини та мова програмування JavaScript для клієнтської частини.

Для реалізації проєкту була обрана мова програмування JavaScript, а саме: бібліотеку ReactJS - для розробки клієнтського інтерфейсу (frontend) та Node.js - для забезпечення серверної (backend) частини проєкту.

2.1. Мова програмування JavaScript

JavaScript - це мова, яка підтримує об’єктно-орієнтований та функціональний стилі програмування, забезпечуючи виконання динамічного оновлення інформації на вебсторінці, анімує зображення, використовує мультимедію тощо [1]. Цю мову можна застосовувати на різних пристроях, що використовують програму Javascript engine.

Зазвичай ця мова використовується в браузерах, які містять різні версії “движків” для компілювання коду.

Нариклад:

- V8 – використовується для Google Chrome та Opera;
- SpiderMonkey – для браузеру Mozila Firefox;
- Nitro – для Safari та інші.

Основи робота “движка” браузеру є досить зрозумілими, але написання такого програмного забезпечення потребує глибоких знань в певних напрямках програмування.

“Движок” працює наступним чином:

- спочатку зчитує текст скрипта (програми);
- перетворює його в мову низького рівня;
- запускає скомпільований код, який працює достатньо швидко [2].

Для виконання даного проєкту було обрано мову TypeScript через її переваги над JS.

TypeScript – це мова, основою якої є JavaScript, але з використанням визначення статичного типу. При компіляції TS перетворюється у JS за допомогою компілятора Typescript або Babel [3]. Даний процес зображений на рисунку 2.1.

Тип – це спосіб обписати форму об’єкта, що дозволяє компілятору перевірити коректність коду.

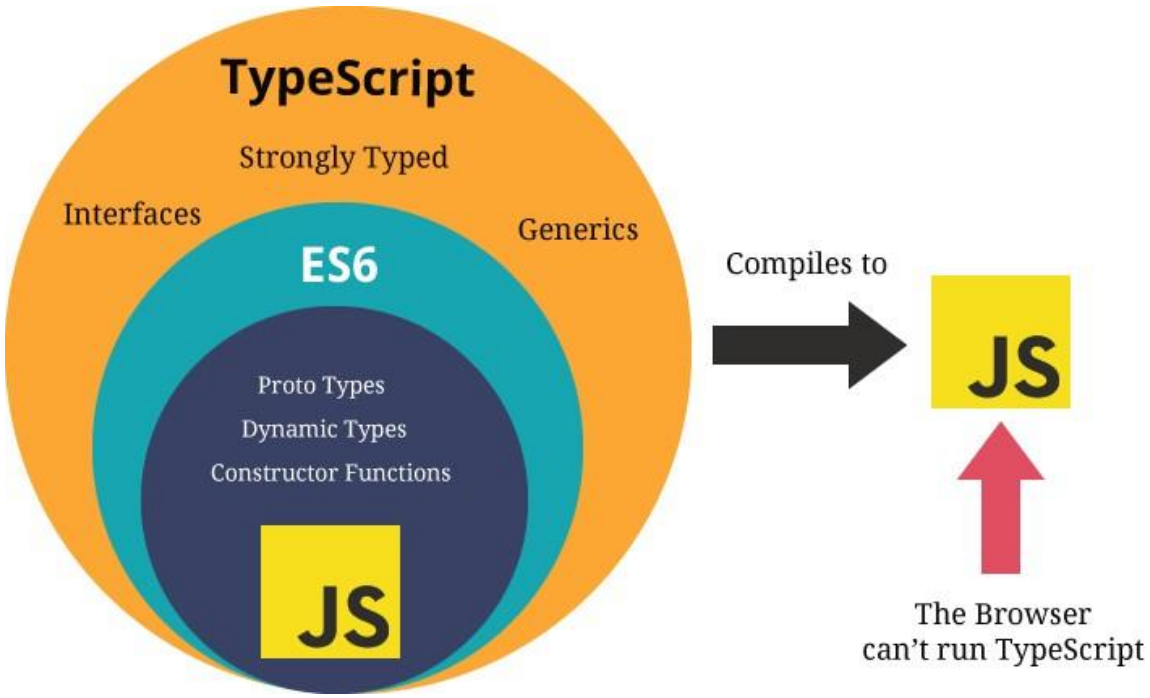


Рис. 2.1 – Процес застосування TypeScript

2.2. Інструмент створення інтерфейсів користувача React

React – бібліотека для мови програмування JavaScript, яка була створена компанією Facebook та використовується при розробці популярних сайтів таких як: Facebook, Twitter, Instagram та інші. Зазвичай, ця бібліотека використовується для створення односторінкових або мобільних додатків, головною задачею якої є відображення змісту вебсторінки [4].

Пакетний менеджер – це інструмент для використання зовнішніх модулів та залежностей, що застосовують для пришвидшення виконання завдання та мінімізації власного коду.

Для застосування даної бібліотеки використовують пакетний менеджер npm або yarn.

Для застосування бібліотеки React використовують файли з розширенням .jsx , що дозволяє застосовувати HTML синтаксис під час описання структури інтерфейсу.

Під час обрання функціоналу для розробки дипломного проєкту головною перевагою цієї бібліотеки стала підтримка віртуальної моделі DOM, що дозволяє оновлювати необхідні частини сторінки без оновлення тих компонентів, які не зазнали змін. Оскільки в додатку присутні обрахунки та аналіз даних, то постійне оновлення моделі є недоречним та займає багато часу. Графічне порівняння зображено на рисунку 2.2.

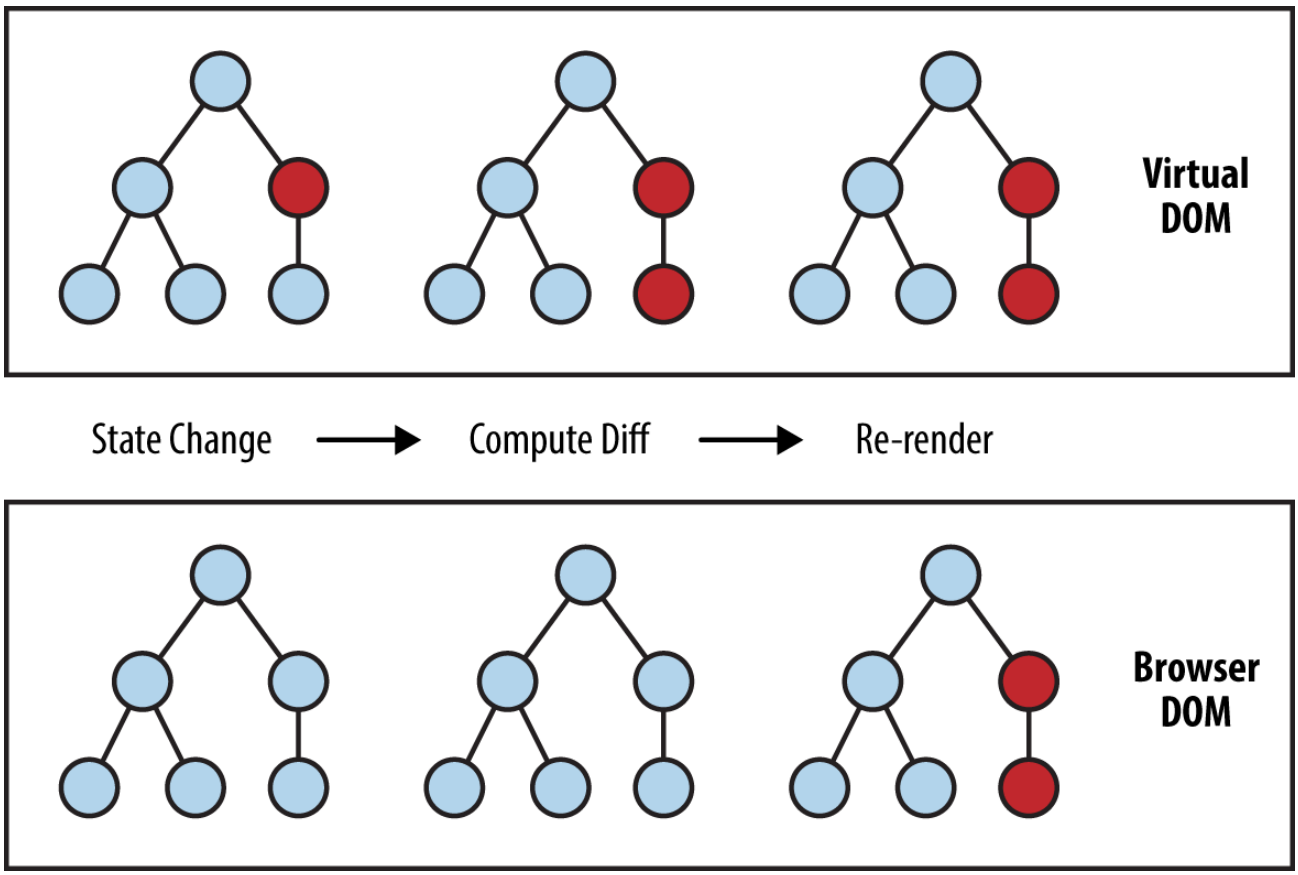


Рис. 2.2 – Порівняння віртуальної DOM з регулярною DOM

Другою перевагою є розбиття коду на невеликі фрагменти, що називаються компоненти, та їх повторне використання.

Компонент – це частина програми, котра забезпечує виконання певного фрагменту інтерфейсу. При необхідності використання однакових елементів, наприклад, “таблиця”, “кнопка”, “поле введення”, а також написання та використання стилів для компонентів такого типу, значно зменшують обсяг коду та дозволяють без окремих зусиль та витрат часу дотримуватися загальної стилістики сайту [4].

Наступна перевага – це наявність додаткових інструментів для розробників, які можна використовувати в різних браузерах, що дозволяє легше відлагоджувати програму та виявляти помилки. Найбільш поширеними є:

- React Developer Tools
- Redux DevTools

Ще однією перевагою є додаткова бібліотека Redux, яка служить сховищем даних і до якої можна приєднатись з будь-якого місця та отримати лише необхідні дані, що прискорює роботу додатку.

Тепер розглянемо фреймворк Angular, який був розроблений компанією Google. На відміну від ReactJS даний фреймворк використовує стандартну модель DOM, що потребує оновлення моделі DOM-дерева для відображення зміненої інформації.

Але головною відмінністю між Angular та React є те, що перший з них є фреймворком, а другий – бібліотекою. Таким чином написаний код використовується по різному. Код бібліотеки викликається та використовується програмістом, а фреймворк, навпаки, викликає код, написаний програмістом, та може включати в себе інші бібліотеки.

Angular та React мають спільні функції такі як:

- використання лише однієї сторінки з подальшим оновленням лише її змісту;
- повторне використання компонентів.

Аналогом бібліотеки Redux для Angular є використання сервісів. Але оскільки структура компонентів є більш складною за React, то отримання інформації з різних рівнів має відмінності.

Критерій	React	Angular
Розробник	Facebook	Google
Визначення	Фреймворк	Бібліотека
Мови	TS	JS, JSX, TSX
DOM	Regular DOM	Virtual DOM
Оновлення інформації	Вся сторінка	Окремі компоненти

Таблиця 2.1 – Порівняльна таблиця даних ресурсів розробки

2.3. Менеджер станів Redux

Redux – менеджер станів, що дозволяє отримувати дані з будь-якого компонента програми [5]. В цьому менеджері станів дані представлені деревом станів, яке доступне лише для зчитування інформації. Для змінення інформації необхідно надіслати action (дію).

Action (дія) – це об’єкт мови програмування JavaScript, котрий описує зміст зміни інформації. Об’єкт мусить мати своє поле type, з типом даних String, та критерії об’єкту при необхідності.

Для створення action використовують генератори дій, що зазвичай оголошуються разом з функцією відправки дії.

Наступної частиною Redux є редуктор (reducer).

Редуктор – це функція, що визначає наступний стан дерева станів в залежності від його попереднього стану та дії, що була застосована. Тобто він повертає новий об’єкт, котрим заміняє попередній.

Store (сховище) – об’єкт, де зберігаються дані додатку. Цей об’єкт має функцію getState(), що повертає стан даних та dispatch(), функція, яка приймає за аргумент action та оновлює стан сховища.

Алгоритм послідовності використання перерахованих елементів зображено на рисунку 2.3.

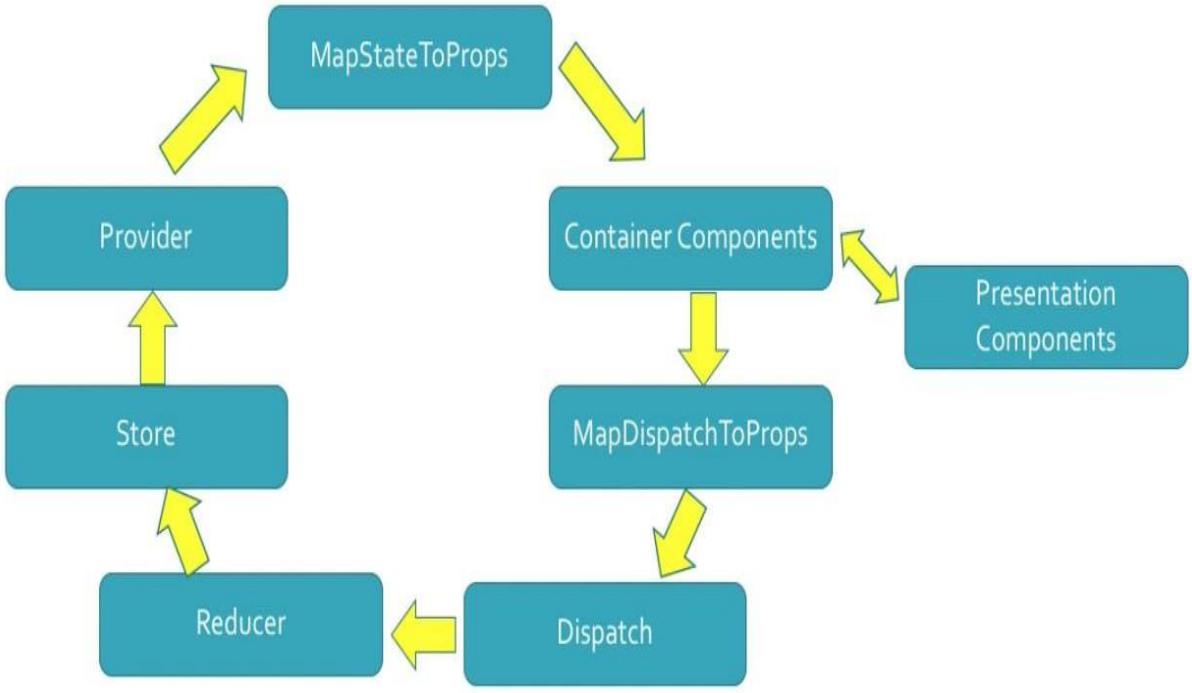


Рис. 2.3 – Алгоритм використання бібліотеки Redux

2.4. Використання середовища NodeJS для реалізації серверної частини програми

NodeJS – це кросплатформне середовище з відкритим кодом виконання JavaScript, яке реалізує код за межами браузера та використовує вже згаданий “движок” V8. Дане середовище дозволяє писати код серверної частини додатків для динамічних вебсторінок та вебдодатків.

Головною перевагою є те, що NodeJS також використовує мову JavaScript для написання коду та спрощує розгортання вебдодатків, оскільки всі сучасні браузери підтримують мову програмування JavaScript, і реалізує парадигму “JavaScript для всього”.

Можна виділити наступні переваги:

- збільшення ефективності розробки через використання єдиної мови програмування. Можливість повторного використання коду;
- наявність об’ємних пакетних менеджерів.

2.5. Вебфреймворк Express

NodeJS має 5 найбільш популярних фреймворків:

- Express.js
- Hapi.js
- Mojito
- Meteor
- Socket.io

Всі вони були розглянуті та проаналізовані. Після отриманих результатів було обрано фреймворк Express.js

Express – це гнучкий фреймворк для середовища NodeJS, що включає масивний набір допоміжних засобів для виконання вебдодатків [6].

При аналізі фреймворку Express були виділені наступні переваги:

- простий, гнучкий та легко піддається масштабуванню, тобто додавання інших вузлів до існуючої системи;
- створення додатків за короткий проміжок часу;
- просте додавання сторонніх ресурсів та програмного забезпечення;
- можливе індивідуальне налаштування [7].

Для використання даного фреймворку необхідно інсталювати його за імпортувати за допомогою команди:

const express = require(‘express’).

2.6. База даних MongoDB.

База даних – це структурована інформація, що зберігається на електронному пристрої та контролюється СУБД. Ці засоби разом з прилеглими до них програмами називаються системою баз даних. Більшість баз даних використовують мову програмування SQL.

SQL – мова програмування для реляційних баз даних для створення запитів, контролю доступу та роботи з даними [8]. Реляційна база даних – це база даних представлена у вигляді двомірної таблиці.

MongoDB – це база даних, представлена у вигляді документо-орієнтованої моделі даних, через що має більшу швидкість за реляційні бази даних. Порівняння SQL DB та MongoDB наведено на рисунку 2.4.

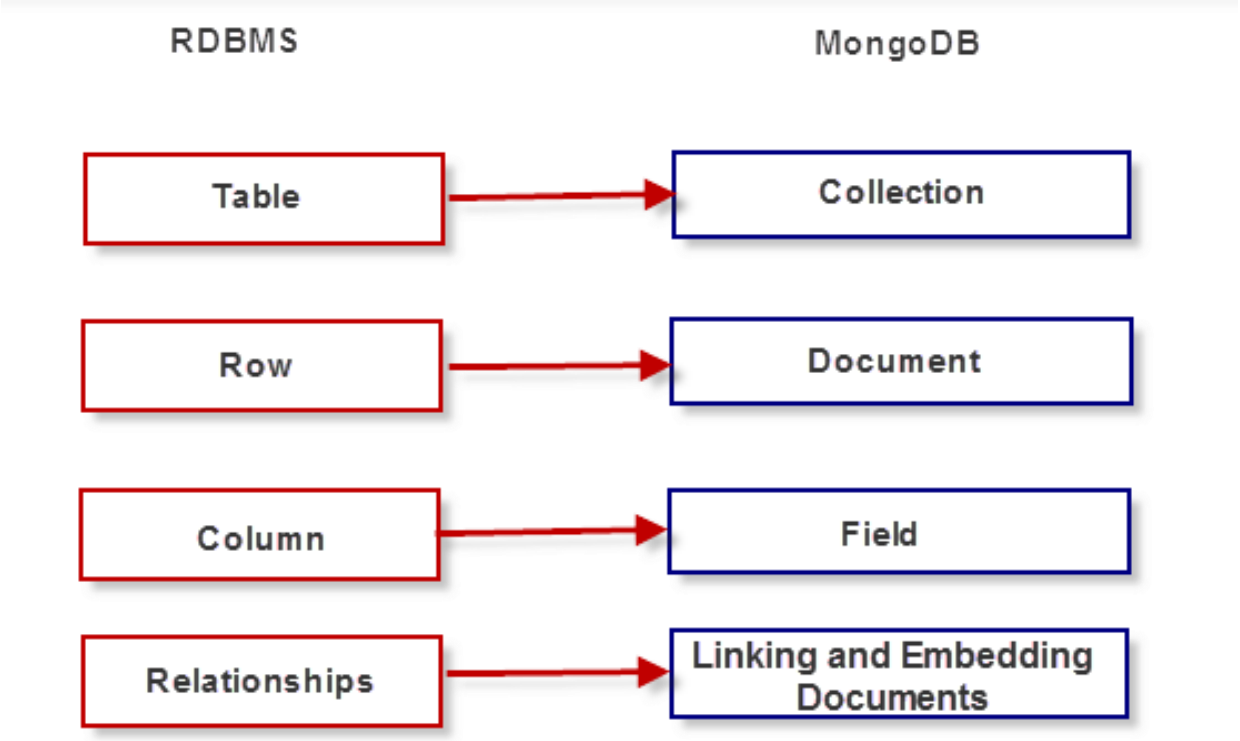


Рис. 2.4 - Порівняння SQL Data Base та MongoDB

Замість таблиць MongoDB використовує колекції файлів для зберігання інформації. Дані безпосередньо зберігаються у JSON файлах як зображено на рисунку 2.5, що дозволяє змінювати структуру з часом.

```
{
    "type": "object",
    "properties": {
        "id": {
            "type": "integer",
            "description": "User ID"
        },
        "first_name": {
            "type": "string",
            "description": "First name"
        },
        "last_name": {
            "type": "string",
            "description": "Last name"
        }
    },
    "required": ["id", "first_name", "last_name"],
    "additionalProperties": false
}
```

Рис. 2.5 – Приклад JSON файлу

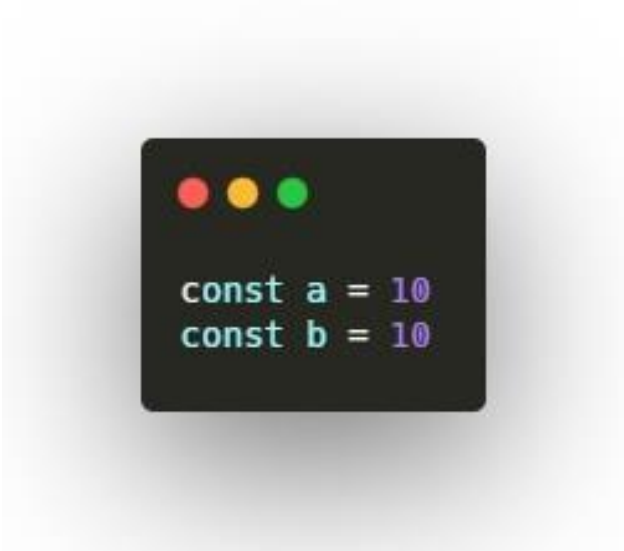
2.7. Пакувальник модулів Webpack.

Для поєднання усіх складових клієнтської частини вебдодатку та перетворення їх у вигляд зрозумілий компілятору використовують постачальники модулів.

Постачальник модулів – це інструмент, за допомогою якого виконується поєднання всіх файлів одного типу в один та відбувається процес мінімізації та шифрування коду.

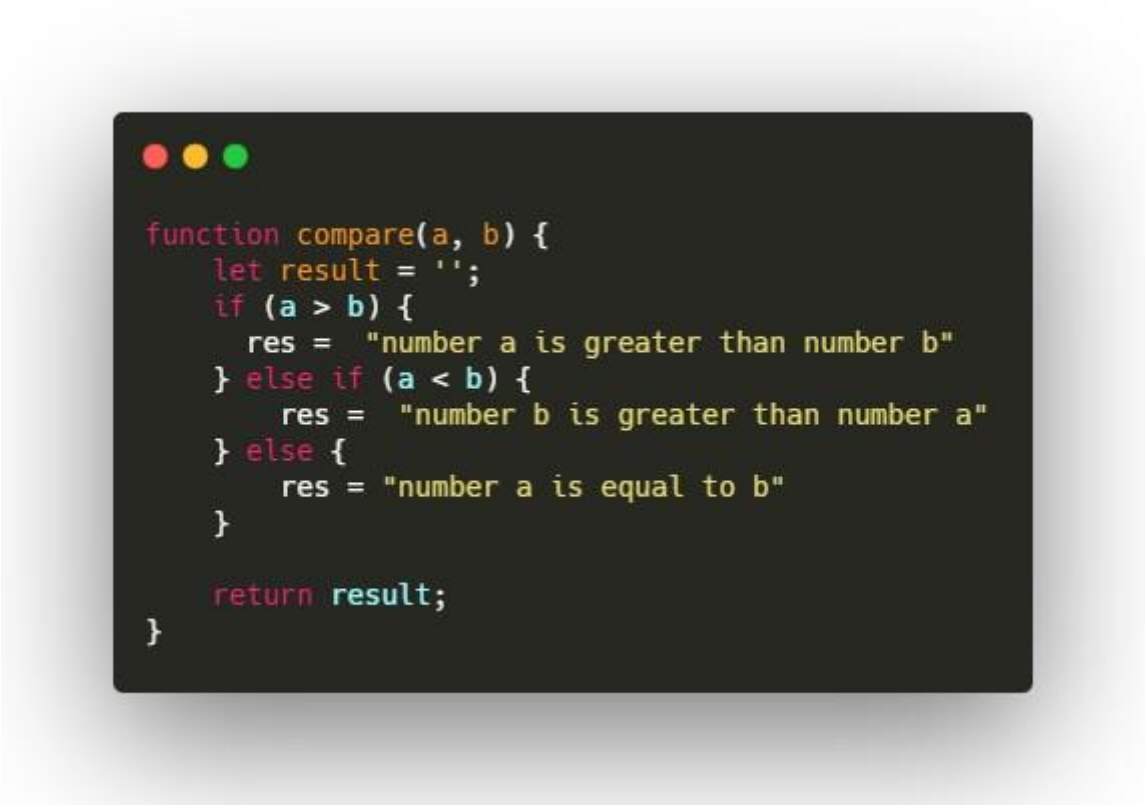
Наприклад, маємо декілька файлів у проекту values.js, function.js та output.js.

Зміст файлів відповідно зображений на рисунках 2.6, 2.7, 2.8



```
const a = 10
const b = 10
```

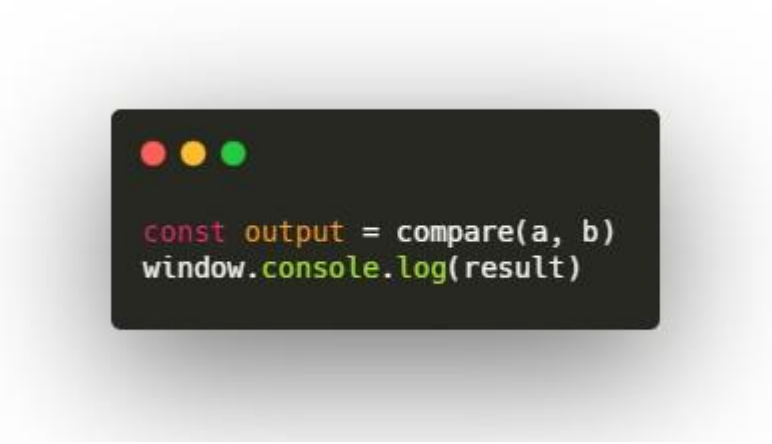
Рис. 2.6 – Зміст файлу values.js



```
function compare(a, b) {
    let result = '';
    if (a > b) {
      res =  "number a is greater than number b"
    } else if (a < b) {
        res =  "number b is greater than number a"
    } else {
        res = "number a is equal to b"
    }

    return result;
}
```

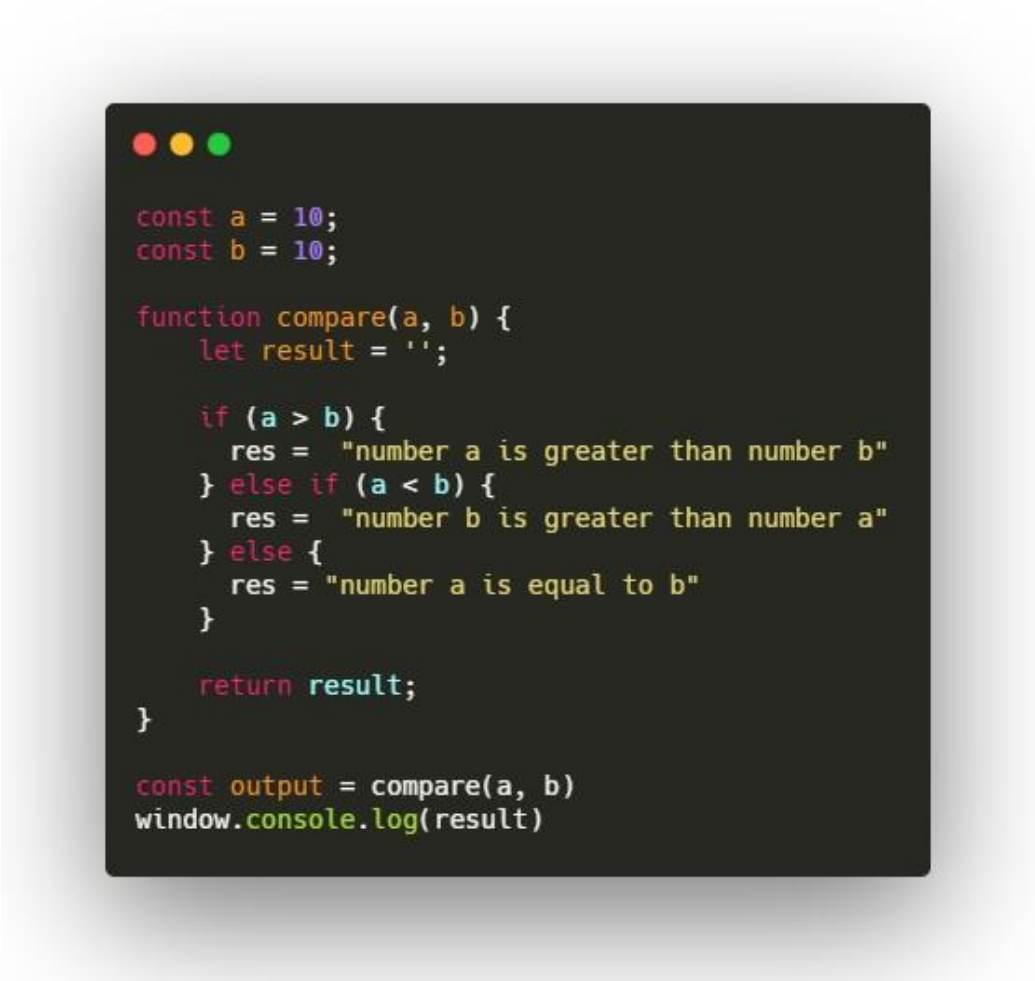
Рис. 2.7 – Зміст файлу function.js



```
const output = compare(a, b)
window.console.log(result)
```

Рис. 2.8 – Зміст файлу output.js

Після компіляції отримаємо один файл із змістом, що складається з усіх інших. Зображено на рисунку 2.9.



```
const a = 10;
const b = 10;

function compare(a, b) {
    let result = '';

    if (a > b) {
      res =  "number a is greater than number b"
    } else if (a < b) {
      res =  "number b is greater than number a"
    } else {
      res = "number a is equal to b"
    }

    return result;
}

const output = compare(a, b)
window.console.log(result)
```

Рис. 2.9 – Зміст файлу результату

Перевагою даного інструменту є запит лише одного файлу, не порушуючи модульного стилю програмування.

Webpack – постачальник модулів, який підтримує формати модулів ES2015, CommonJS та AMD. Також підтримує механізм роботи для файлів CSS з *@import* та url() за допомогою css-loader.

Процес перетворення файлів за допомогою Webpack зображено на рисунку 2.10.

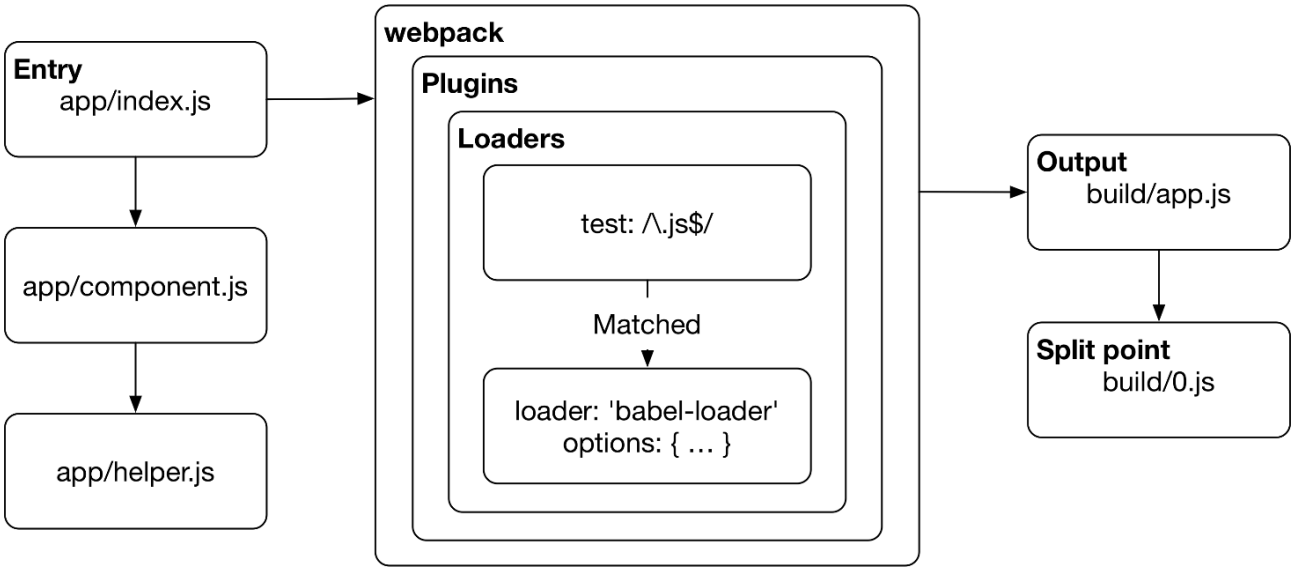


Рис. 2.10 – Процес виконання Webpack

2.8. Предпроцесори CSS

Предпроцесор CSS – це мова написання стилів, які схожі на CSS, котрі включають в себе можливості, що не працюють напряму у браузері, такі як:

- вкладення – дозволяє наглядно бачити належність стилів одного елементу до іншого (рисунок 2.11);
- використання символу амперсанд «&» - дозволяє не друкувати повну назву батьківського елементу (рисунок 2.12);

- міксини – використовується задля запобіганню повторення коду стилів (рисунок 2.13);
- умови;
- цикли;
- функції;

Також використання CSS-препроцесорів має наступні переваги:

- прискорює написання стилів
- спрощує оновлення коду

Недоліком є використання додаткового програмного забезпечення задля компіляції коду в «чистий» CSS.

```
.promo {
  padding: 1em;

  h1 { // после компиляции превратится в .promo h1 {...}
    font-size: 48px;
  }

  @media (min-width: 1024px) { // после компиляции «вывернется»
    padding: 2em;
  }
}
```

Рис. 2.11 – Застосування вкладення

```
.btn {
  color: #000;

  &:hover { // после компиляции превратится в .btn:hover {...}
    color: #808080;
  }

  &__icon { // после компиляции превратится в .btn__icon {...}
    position: absolute;
  }

  .promo & { // после компиляции превратится в .promo .btn {...}
    margin-left: 30px;
  }
}
```

Рис. 2.12 – Застосування амперсанду

```
// Создание примеси
@mixin clearfix {

  &:after {
    content: "";
    display: table;
    clear: both;
  }
}

// Применение примеси
.promo__wrapper {
  @include clearfix;
  padding: 1em;
}
```

Рис 2.13 – Застосування міксинів

3. ВЕБДОДАТОК ДЛЯ ОБЛІКУ КОШТІВ ТА ПЛАНУВАННЯ ВИТРАТ

3.1. Вебдодаток. Його переваги, складові та види.

Вебдодаток – це програма, яка виконується та зберігається на віддаленому сервері та передаються через Інтернет в інтерфейс браузера. Для виконання вебдодатку необхідно:

1. вебсервер;
2. сервер додатків;
3. база даних.

Перевагами вебдодатків є:

1. доступ великої кількості користувачів до однієї програми;
2. не потребують інсталяції;
3. можна використовувати той самий додаток на різних платформах;
4. доступ з декількох браузерів.

Вебсервер – це ПЗ, яке надає доступ для кінцевого користувача через доменні імена вебсайтів, які зберігаються на сервері, за допомогою URL-адреси та HTTP протоколу. Якщо розглядати вебсервер, яка апаратне забезпечення, то вебсервер – це комп'ютер, який зберігає дані вебсайтів, таких як HTML документи, зображення, таблиці стилів та логіку вебсайтів.

Вебсервери в свою чергу поділяються на данамічні та статичні. Головна різниця полягає у тому, динамічний спочатку завантажує необхідну інформацію і лише потім надсилає ці дані, статичний лише надсилає існуючу інформацію.

Вебдодатки поділяються на:

- статичні – додатки виконані за допомогою HTML та CSS. Містять відносно малу кількість інформації. Можливе використання бібліотеки jQuery, Ajax запитів та використання анімаційних засобів (відео, банери, GIFs);

- динамічні – додатки, що технічно на порядок вище за статичні. Використовують базу даних для завантаження змісту сторінки. Зазвичай мають адмінпанелі для модерації та оновлення наповнення вебдодатку;
- додатки для продажу - додатки, що має складний процес виконання, оскільки додаток мусить виконувати електронні платежі, що можна здійснювати завдяки електронним способам оплати. Такі сайти завжди мають адмінпанель для оновлення та додавання нових товарів;
- портали – вебдодатки з певною кількістю розділів. Включають в себе різноманітні сервіси: пошта, чати, реєстраційні форми та інше [9].

Згідно до цих завдань, сайт поділяються на такі типи:

- Рекламні вебсайти - сайти можуть створюватися виключно в рекламно-промоутерських цілях. Такі сайти безпосередньо не займаються продажем. Їх завдання полягає в донесенні до цільової аудиторії рекламної інформації, і створюються вони з розрахунку на певне коло товарів або послуг. Зазвичай такі сайти створюють з використанням великої кількості графіки, flash-анімації. Для залучення клієнтів на сайт використовують ігрові й розважальні методи;
- Вебсайти продавці - для таких сайтів характерна наявність описового рекламного матеріалу для товарів або послуг, каталог даних товарів або послуг, інформації про фірму-продавця, а також контактна інформація. Додаткові сервіси, такі, як корисна інформація, зручність замовлення через сайт у поєднанні із грамотною розкруткою, можуть зробити веб-сайт привабливим для сторонніх рекламодавців [9];

- Вебсайти «альтруїсти» - інформаційні вебсайти, які надають деякі безкоштовні сервіси, теж потрібно обслуговувати, розвивати, а отже, вкладати в них кошти. Але проекти, які не приносять прибуток, довго не живуть, тому для таких вебсайтів характерне заробляння грошей або на рекламі, або на зборі статистичних даних. На таких сайтах дуже часто пропонують зареєструватися, аби отримати маленький додатковий сервіс;
- Останній тип вебсайтів — це вебсайти для підтримки клієнтів. Як правило на таких сайтах розміщують оновлення для продуктів, новини; якщо йдеться про сайт банку, то це може бути інформація про зміни у системі управління засобами клієнта. Ці інтернет-ресурси є рекламою фірми, товару та інше [9].

Також сайти поділяють за доступністю сервісів:

- відкриті – усі сервіси повністю доступні для будь-яких відвідувачів;
- напіввідкриті – для доступу потрібно зареєструватися;
- закриті – повністю закриті службові сайти організацій, корпоративні сайти, сайти приватних осіб. Такі сайти доступні для вузького кола людей. Доступ нових людей можливий через запрошення.

Ще одним критерієм для розподілення сайтів за фізичним розташуванням.

Якщо сайт доступний користувачам з Інтернету, він вважається зовнішнім, натомість сайт, котрий відвідати здатні лише користувачі локальної мережі, є внутрішнім. Прикладами внутрішнього сайту можуть бути корпоративний сайт підприємства або сайт приватної особи в локальній мережі провайдера [9].

Ще одним критерієм розподілення вебсайтів є їх призначення.

Маємо наступні види:

- бізнес-сайти – сайти, що містять інформацію про компанії та їхні послуги, здійснюють функцію електронної торгівлі;
- інформаційні сайти – призначені для інформування відвідувачів, поширення новин, тематичні сайти, енциклопедії, словники тощо;
- сайти соціальних мереж – інтерактивні багатокористувацькі вебсайти, які наповнюються самими учасниками мережі. Сайт являє собою автоматизоване соціальне середовище, що дозволяє спілкуватися групі користувачів, об'єднаних спільним інтересом;
- вебпортали – універсальні сайти, через які можна вийти на інші ресурси Інтернету;
- сайти сервісів – сайти служб, які існують у мережі Інтернет, зокрема, сайти пошукових служб (Google, Bing), поштові сайти, вебфоруми, он-лайнові сховища даних (Skydrive), сайти служб онлайнового документообігу (Google Docs), зберігання та обробки фотографій (Picnik, ImageShack, Panoramio, Photobucket), зберігання відео (You Tube) [10].

Детальний розподіл вебсайтів за призначенням та залежності від технології зображено на рисунку 3.1.

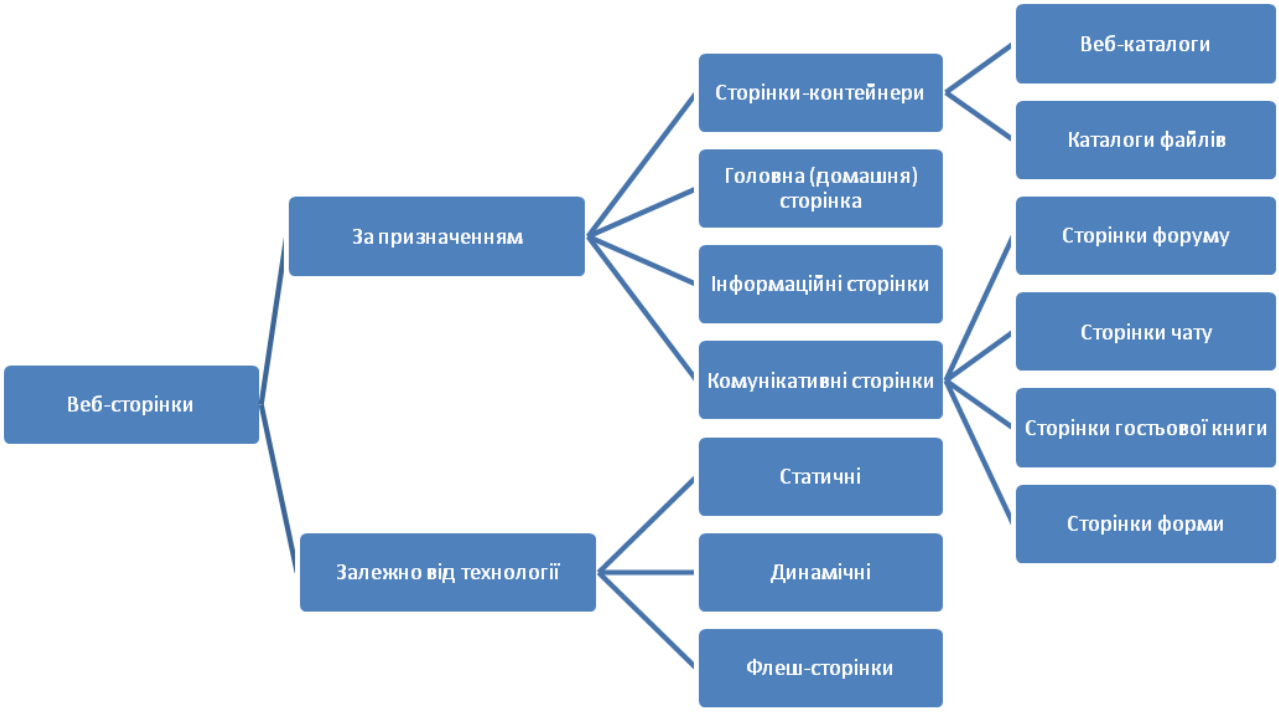


Рис. 3.1 – Типізація вебсайтів

3.2. Розробка дизайну та архітектури

Для реалізації вебдодатку обрана дворівнева архітектура, яка відображена на рисунку 3.2.

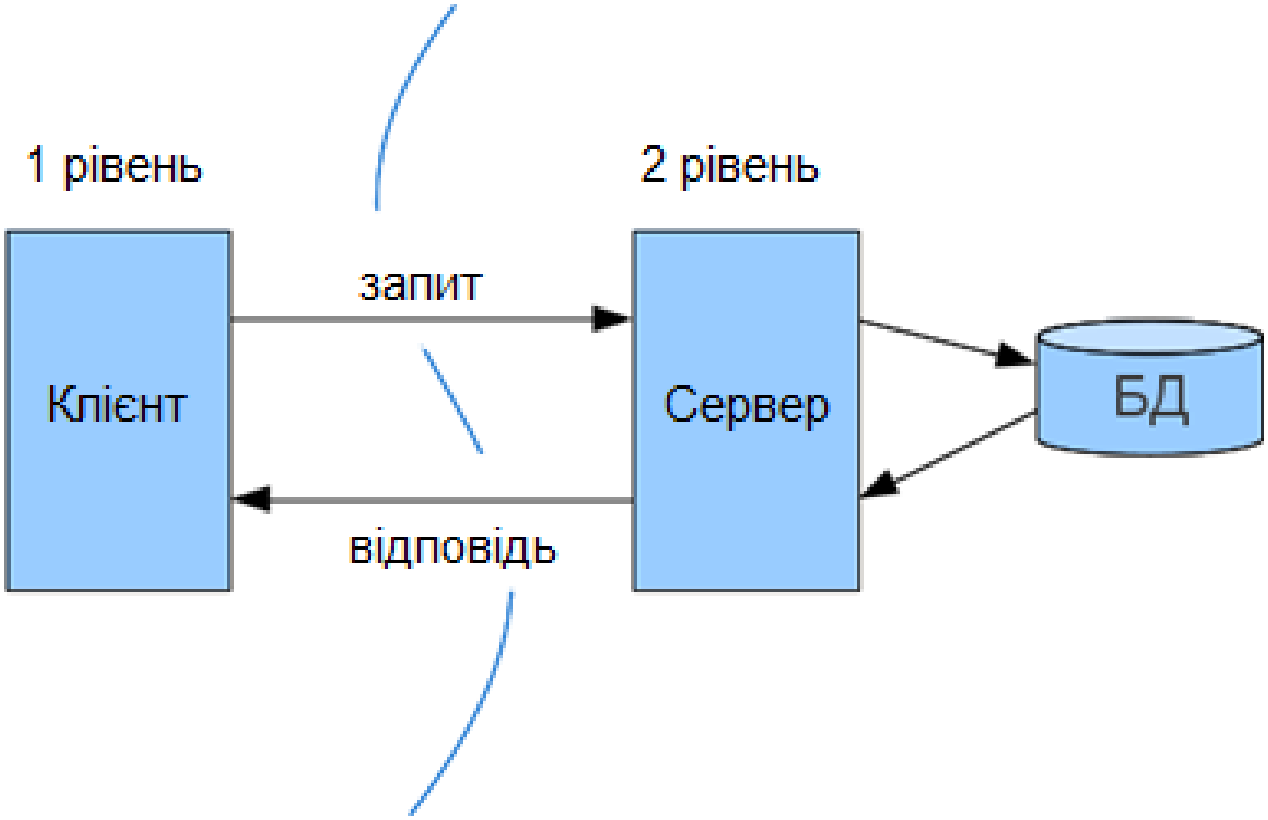


Рис. 3.2 – Дворівнева клієнт-сервер архітектура

При архітектурі «клієнт-сервер» використовуються принципи взаємодії, при яких сервери - вузли-постачальники структур даних та функцій, клієнт – вузол-споживач цієї інформації.

Практичні реалізації даного стандарту мають назву клієнт-серверні технології, кожна з яких створює та використовує правила – протоколи взаємодії. Ці протоколи встановлюють правила взаємодії між клієнтом та сервером.

Дворівнева архітектура отримала свою назву через розподілення трьох базових компонентів (клієнту, серверу та бази даних) між двома вузлами, що за складністю задовольняє даний вебдодаток.

Дворівнева архітектура використовується в клієнт-серверних системах, де сервер не викликає сторонні мережеві програми і не звертається до сторонніх ресурсів для виконання будь-якої частини запиту. Тобто сервер відповідає на клієнтські запити безпосередньо і в повному обсязі, при цьому використовуючи тільки власні ресурси

3.3. Колекції серверної частини програми

Для серверної частини програми розроблено наступні колекції для баз даних.

_id	Унікальний ідентифікатор користувача
Name	Ім’я користувача
Email	Пошта користувача
password	Пароль користувача
Date	Дата створення користувача

Табл. 3.1 – Структура документа користувача.

_id	Унікальний ідентифікатор профілю
user_id	Унікальний ідентифікатор користувача
user_name	Ім’я користувача
email	Пошта користувача
isLoggedIn	Статус аутентифікації користувача

total_profit	Доступна кількість грошей для витрат в поточному місяці
total_expense	Загальна кількість витрат в поточному місяці

Табл. 3.2 – Структура документа профілю.

_id	Унікальний ідентифікатор прибутку
user_id	Унікальний ідентифікатор користувача
Amount	Розмір прибутку
Date	Дата додання прибутку до бази даних

Табл. 3.3 – Структура документа прибутку.

_id	Унікальний ідентифікатор витрати
user_id	Унікальний ідентифікатор користувача
Amount	Розмір витрат
Type	Тип витрат для класифікації
Date	Дата додання прибутку до бази даних

Табл. 3.4 – Структура документа витрати.

_id	Унікальний ідентифікатор цілі
user_id	Унікальний ідентифікатор користувача
cost	Розмір витрат
type	Тип витрат для класифікації
date	Дата додання прибутку до бази даних
already_alocated	Виділена кількість грошей
Finish_date	Кінцева дата для досягнення цілі

Табл. 3.5 – Структура документа цілі.

3.4. Реалізація аутентифікації для користувача

Під час реєстрації користувача створюється дві колекції: користувач та профіль. Колекція користувача використовується для авторизації, а профіль для подальшого використання у вебдодатку.

Для реєстрації у додатку користувача необхідно ввести своє ім’я, адресу електронної пошти та пароль, а також підтвердити обраний пароль в полі підтвердження паролю, як зображено на рисунку 3.3.

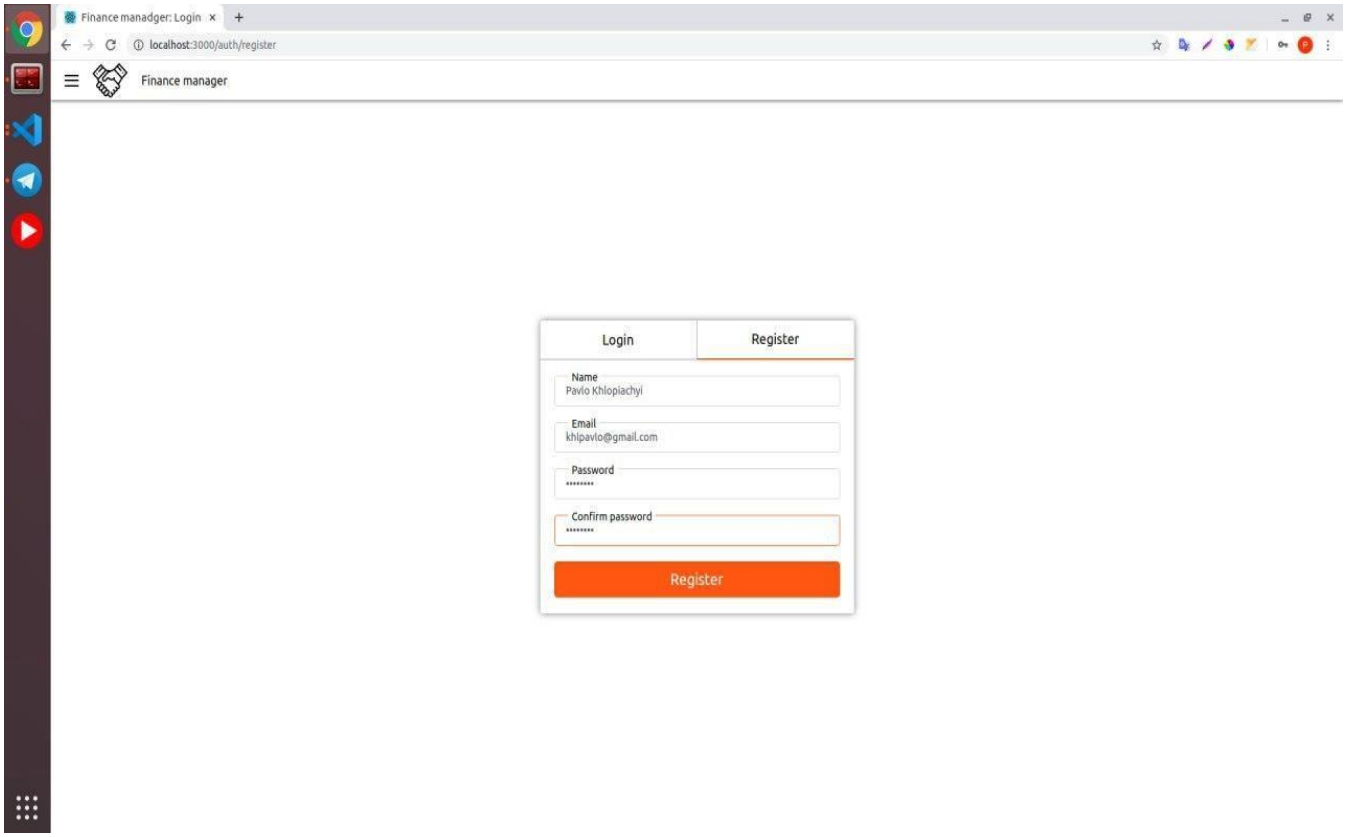


Рис 3.3 – Інтерфейс реєстрації користувача

Під час запиту до серверу відбувається перевірка введених даних на коректність за допомогою бібліотеки Joi, яка є найпоширенішою та найпотужнішою системою перевірки на коректність для мови програмування JavaScript.

Після успішної реєстрації клієнт буде автоматично переадресований на сторінку авторизації, яка зображена на рисунку 3.4.

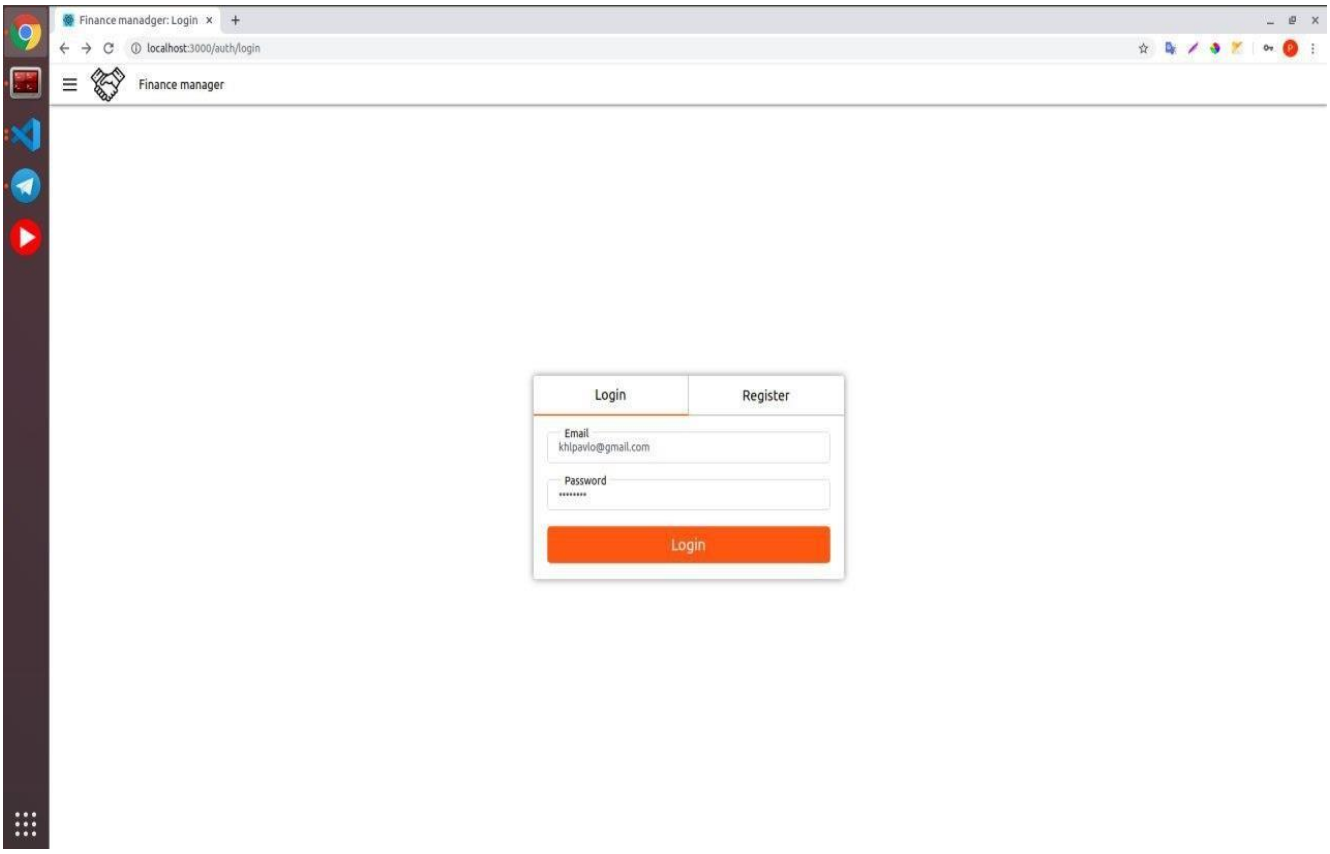


Рис 3.4 – Інтерфейс авторизації користувача

JWT – це відкритий стандарт для компактного та автономного способу передачі між сторонами [11]. У даному випадку це клієнт та сервер.

Під час авторизації створюється два JWT токени для контролю доступу до певних даних та для контролю тривалості сесії, які зберігаються у LocalStorage. Токен такого типу складається з трьох частин, які зображені на рисунку 3.5.

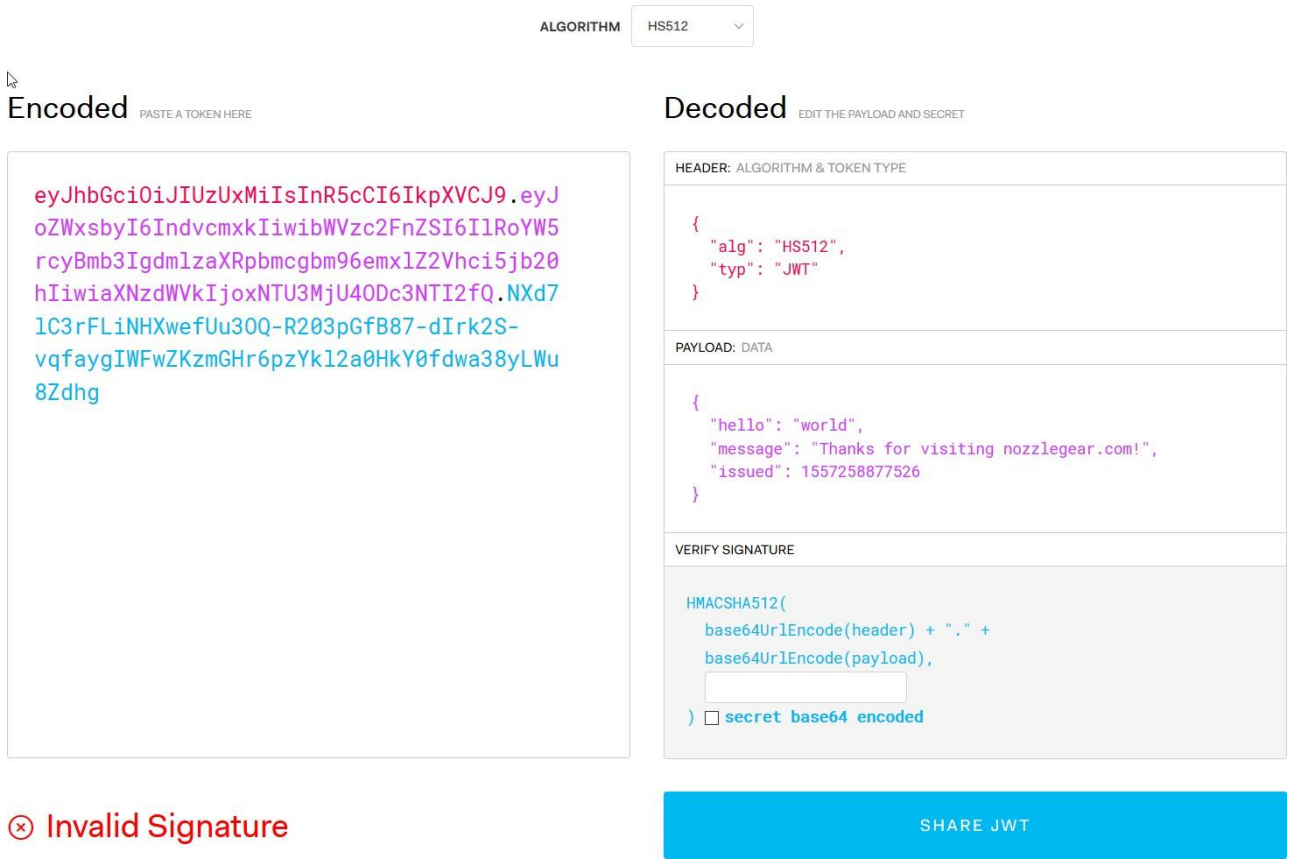


Рис. 3.5 – Складові частини JWT токена

Перша частина JWT токену – це header, який зберігає в собі:

- тип алгоритму;
- тип токену.

Друга частина – тіло токену. Зберігає закодовані дані.

Остання частина – підтвердження токену. Містить алгоритми та функції для кодування та розкодування інформації, яка зберігається у тілі JWT.

LocalStorage – тип веб-сховища, що за допомогою JavaScript дозволяє вебсайтам зберігати та отримувати інформацію без кінцевої дати видалення інформації [12].

Тобто навіть після закриття браузеру дані будуть збережені. Єдиний шлях видалити ці дані, це очистити кеш браузера вручну.

Було обрано саме це місце замість Redux для зберігання токенів через те, що при перезавантаженні сторінки Redux не зберігає жодної інформації.

Отже, користувач має два токени, перший – це access_token з обмеженим періодом дії, котрий відповідає за доступ до певної інформації. Другий токен – refresh_token, який не має терміну дії та зберігається окремо на сервері. Він слугує прапорцем для перевірки авторизації користувача та оновлює access_token.

Алгоритм дій під час API запиту:

- під час авторизації користувач отримує два JWT токени;
- користувач робить запит на сервер, де передає два токени та іншу необхідну інформацію;
- сервер перевіряє чи правильний access_token;
- при коректності цього токену застосовує refresh_token для оновлення access_token, щоб не закінчився термін дії access_token;
- якщо refresh_token не коректний, то користувачу перекривається доступ до даних і йому необхідно пройти авторизацію повторно.

Цей алгоритм графічно відображений на рисунку 3.6.

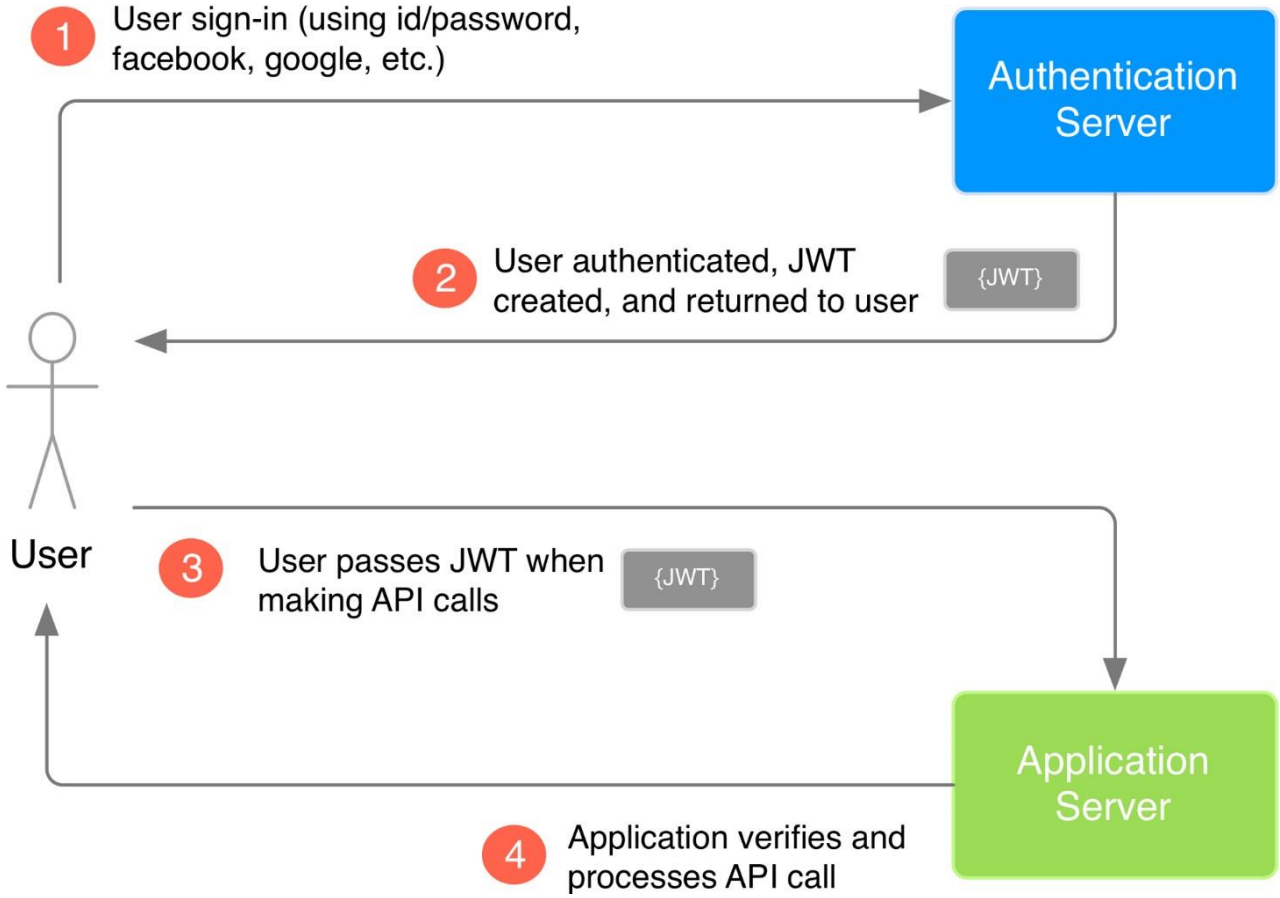


Рис. 3.6 – Алгоритм використання JWT токену

3.5 Розробка клієнтської частини додатку

Після авторизації клієнт опиняється на сторінці його профілю, де відображена інформація:

- ім’я та електронну адресу користувача;
- діаграма співвідношення витрат до прибутку за останній місяць;
- діаграма співвідношення витрат за категорією.

Також користувач отримує змогу використовувати приватні сторінки, що зображені як на рисунку 3.7. та відповідають наступній інформації:

- профіль;
- сторінка прибутків;
- сторінка витрат;
- сторінка цілей.

Рис. 3.7 – Меню авторизованого користувача

У випадку коли для будь-якої таблиці немає записів, то як на рисунку 3.8 зображено таблицю з відповідною інформацією

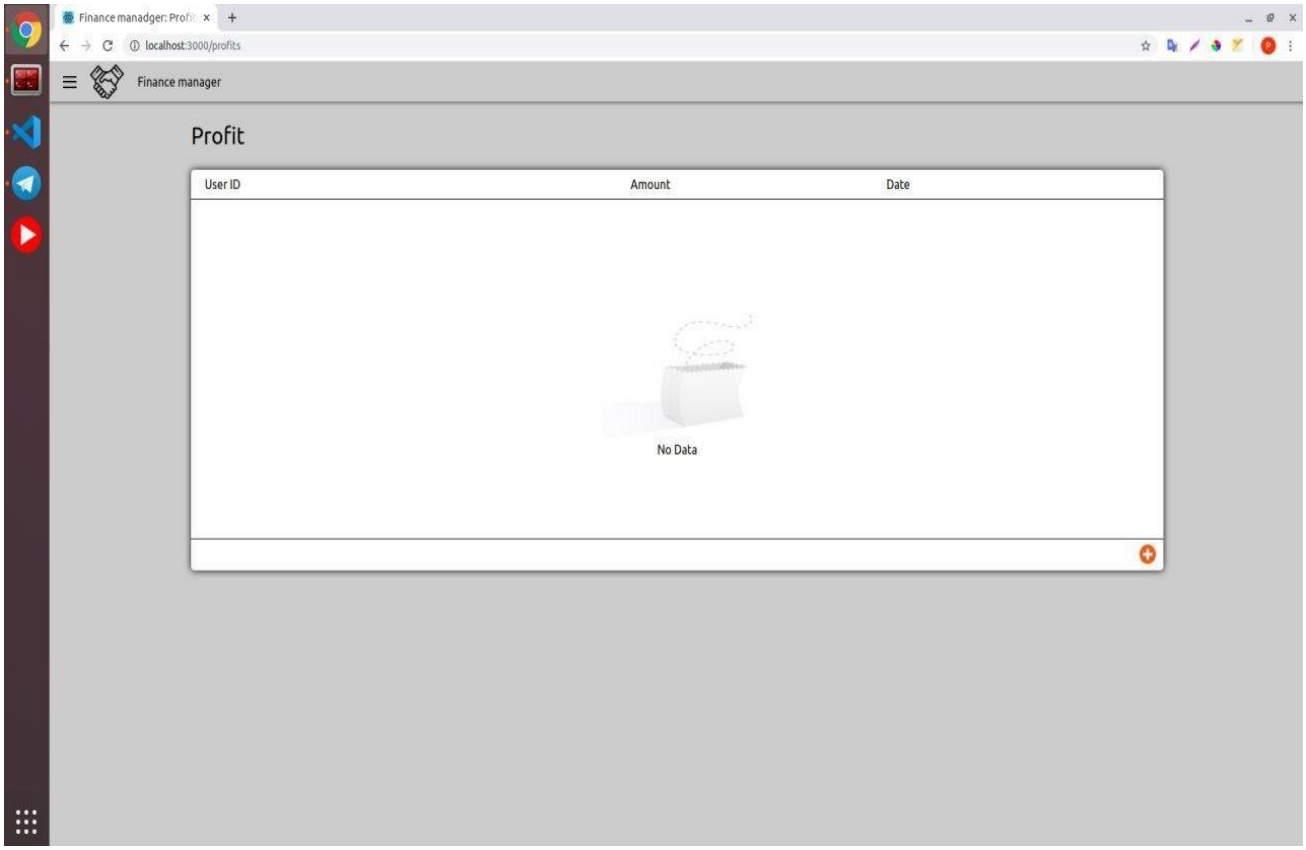


Рис. 3.8 – Інтерфейс таблиці без даних

На сторінці витрат знаходиться таблиця, на якій зображено усі витрати користувача (рисунок 3.9), які він виконував з моменту реєстрації в додатку, а також можливість додавання нової витрати (рисунок 3.10).

Для додавання нового прибутку необхідно лише ввести його величину та надіслати запит.

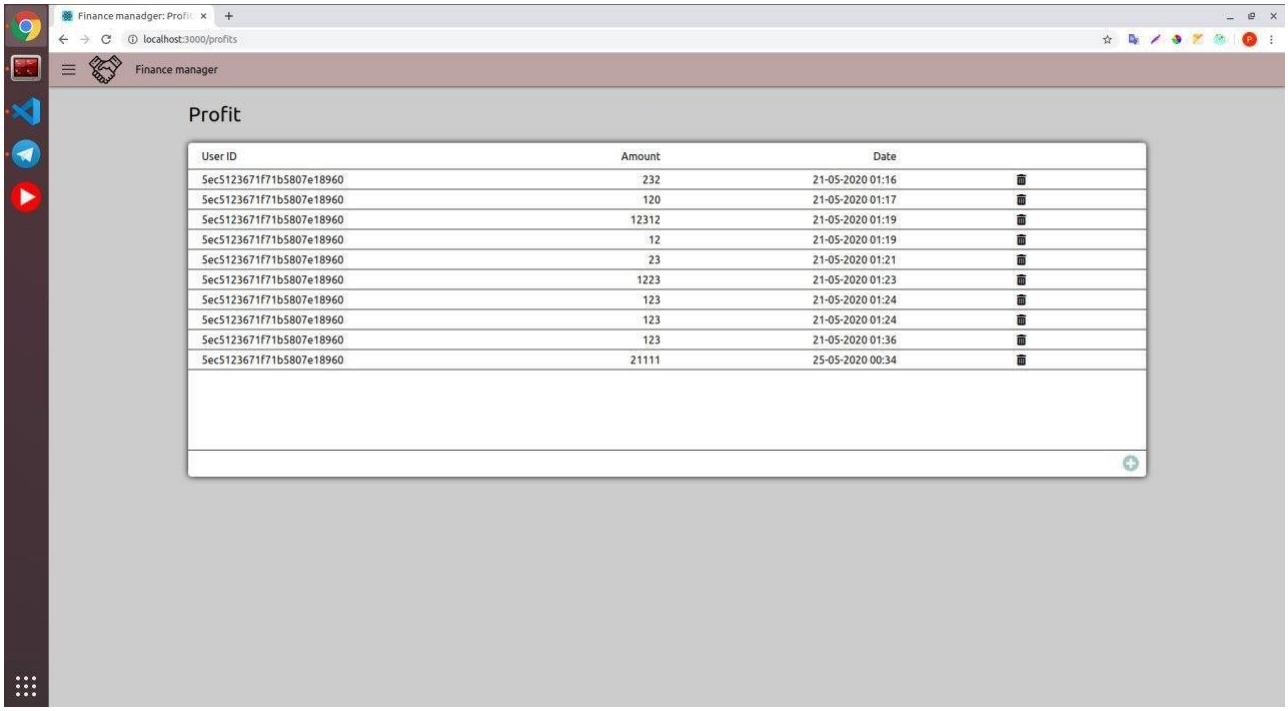


Рис. 3.9 – Інтерфейс прибутків користувача

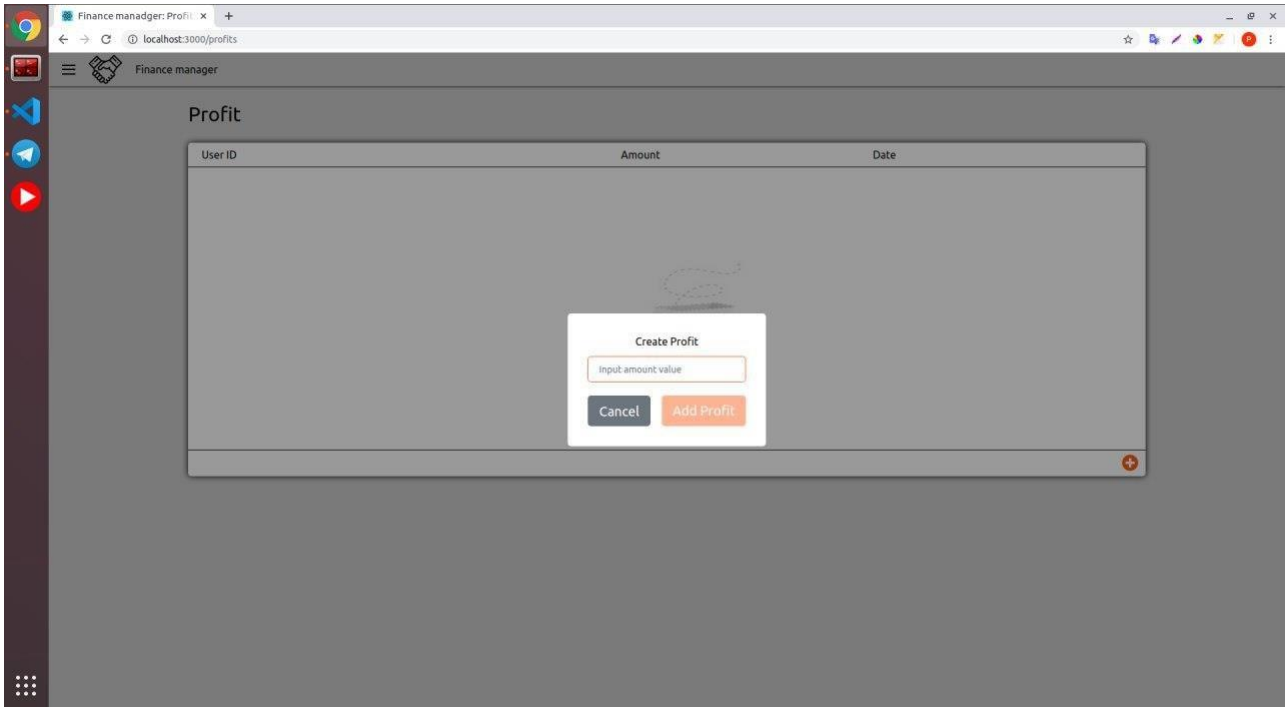


Рис. 3.10 – Інтерфейс додавання прибутку

Для додавання витрати необхідно ввести величину витрати та її тип для подальшої класифікації. Типи витрат зображені на рисунку 3.11. Приклад створення витрати вказано на рисунку 3.12.

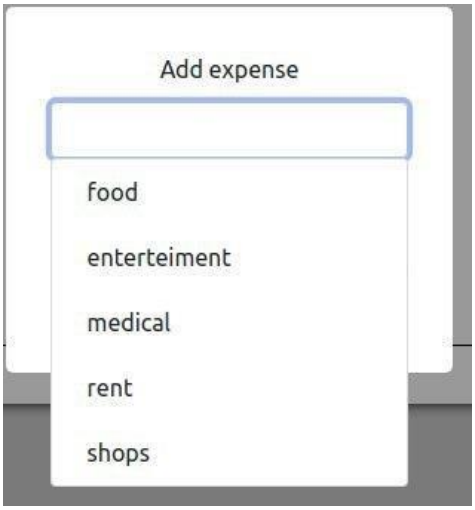


Рис. 3.11 – Типи витрат

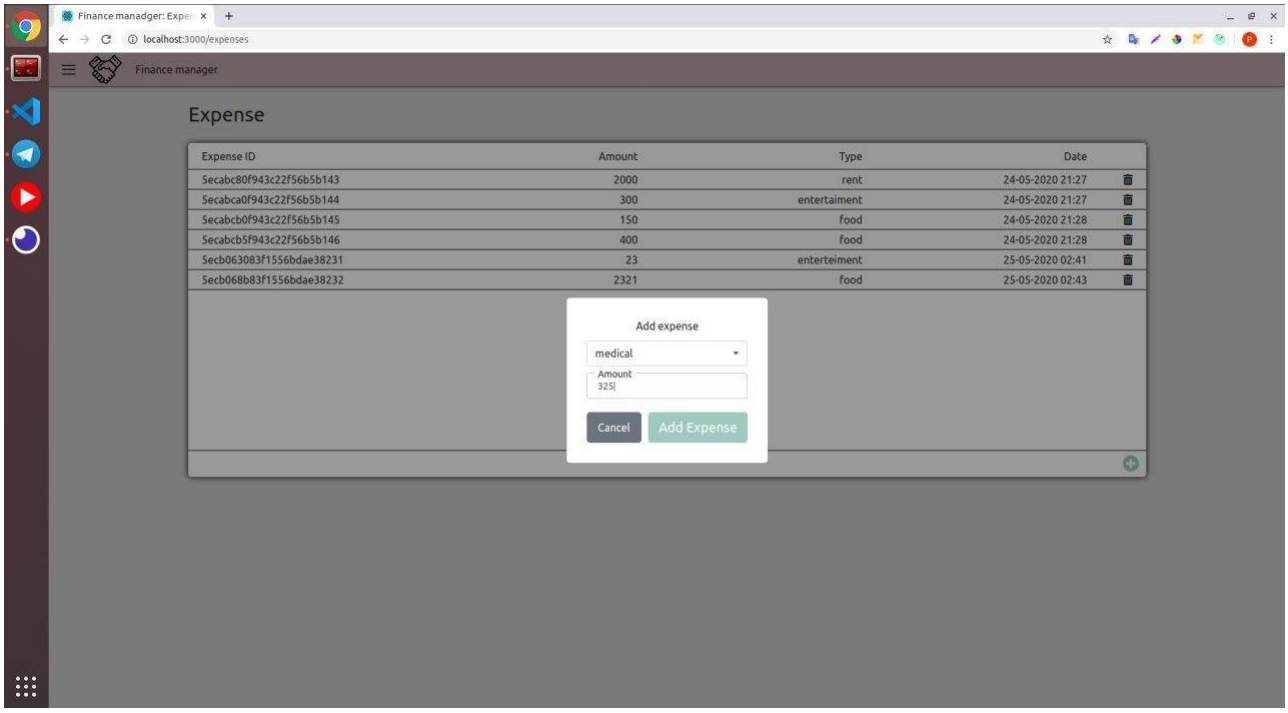


Рис. 3.12 – Інтерфейс створення нової витрати

Після успішного додання витрати, вона додається до таблиці витрат, що зображено на рисунку 3.13.

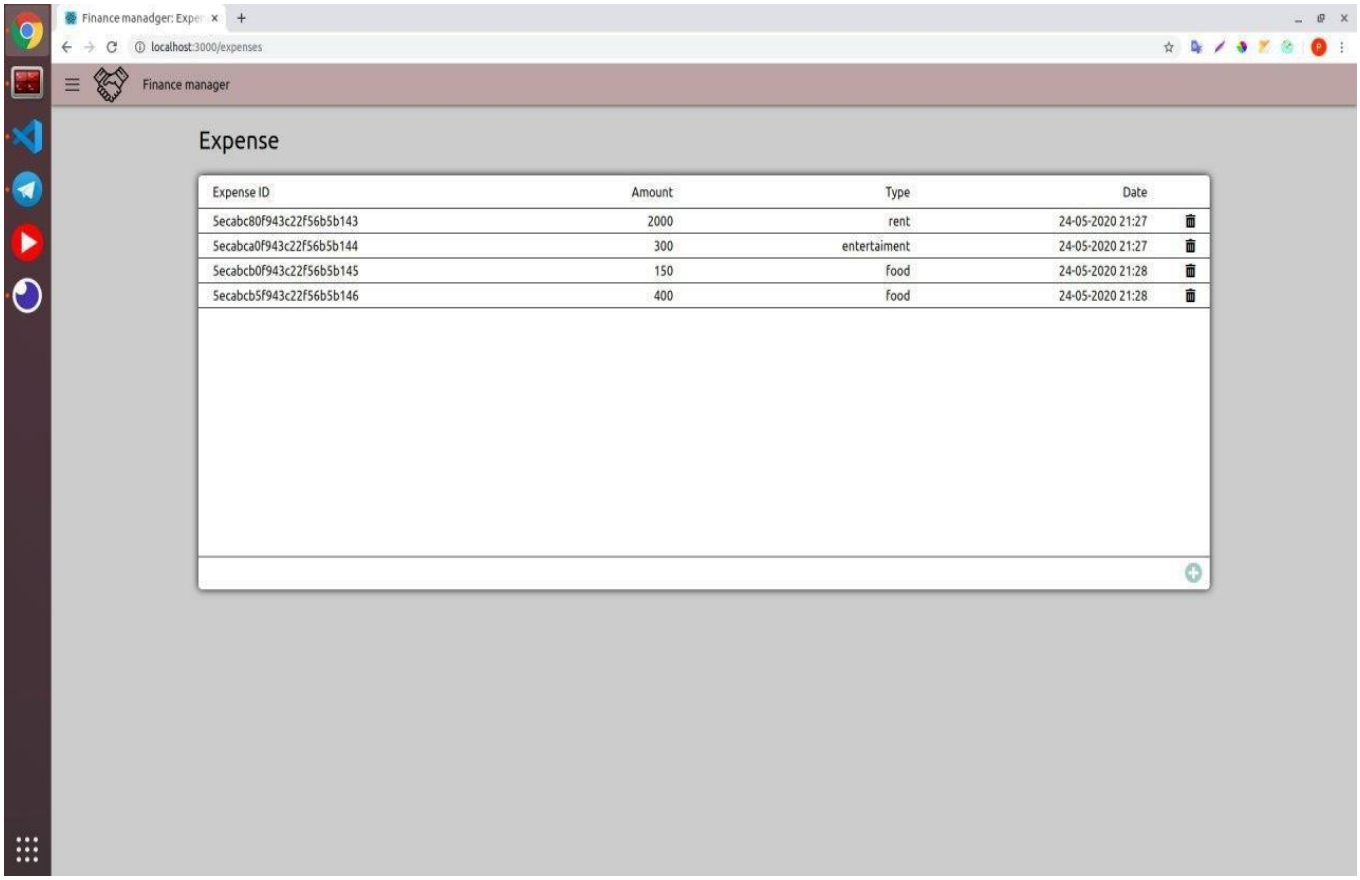


Рис. 3.13 – Інтерфейс списку витрат користувача

Для додання нової цілі необхідно вказати її назву, вартість та дату, на яку заплановано покупка обраного елементу, як зображено на рисунку 3.14.

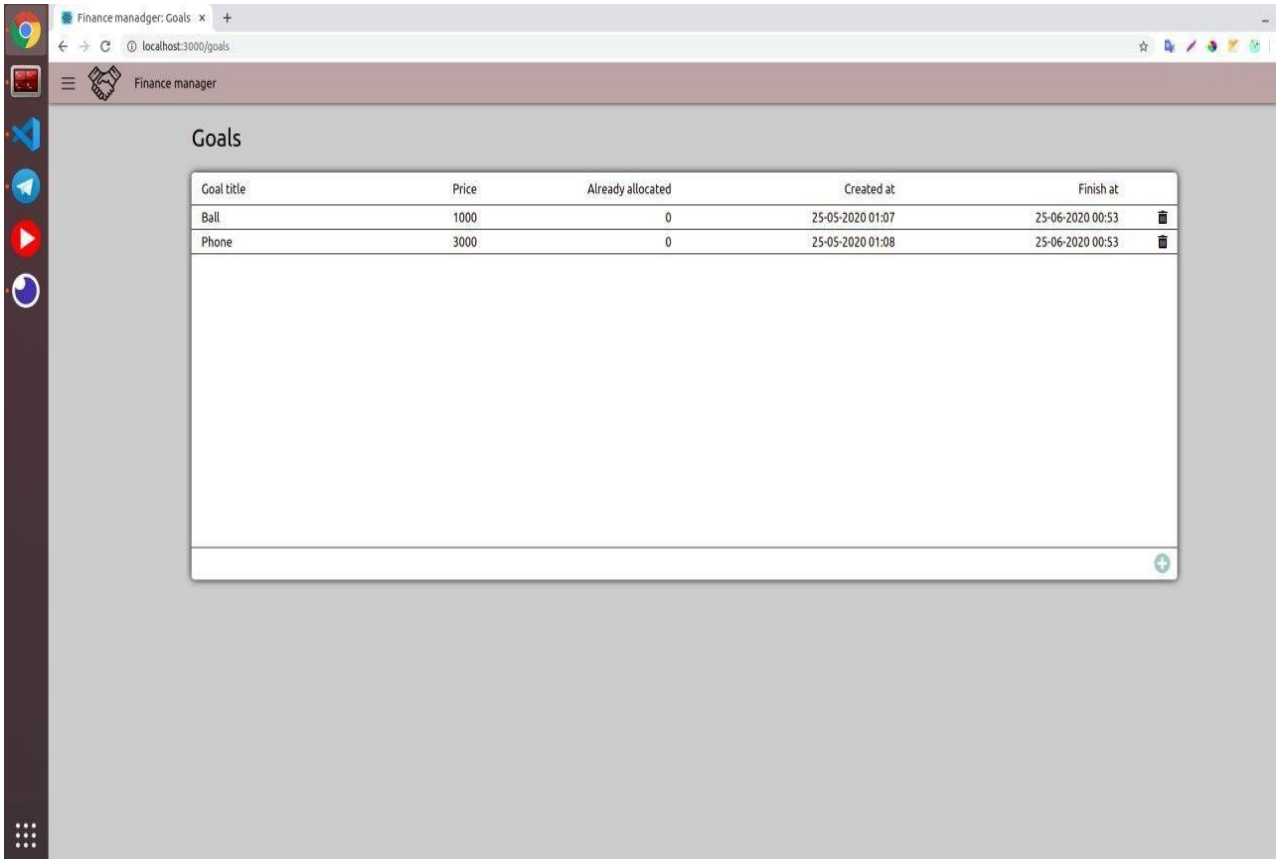


Рис. 3.14 – Інтерфейс списку цілей користувача

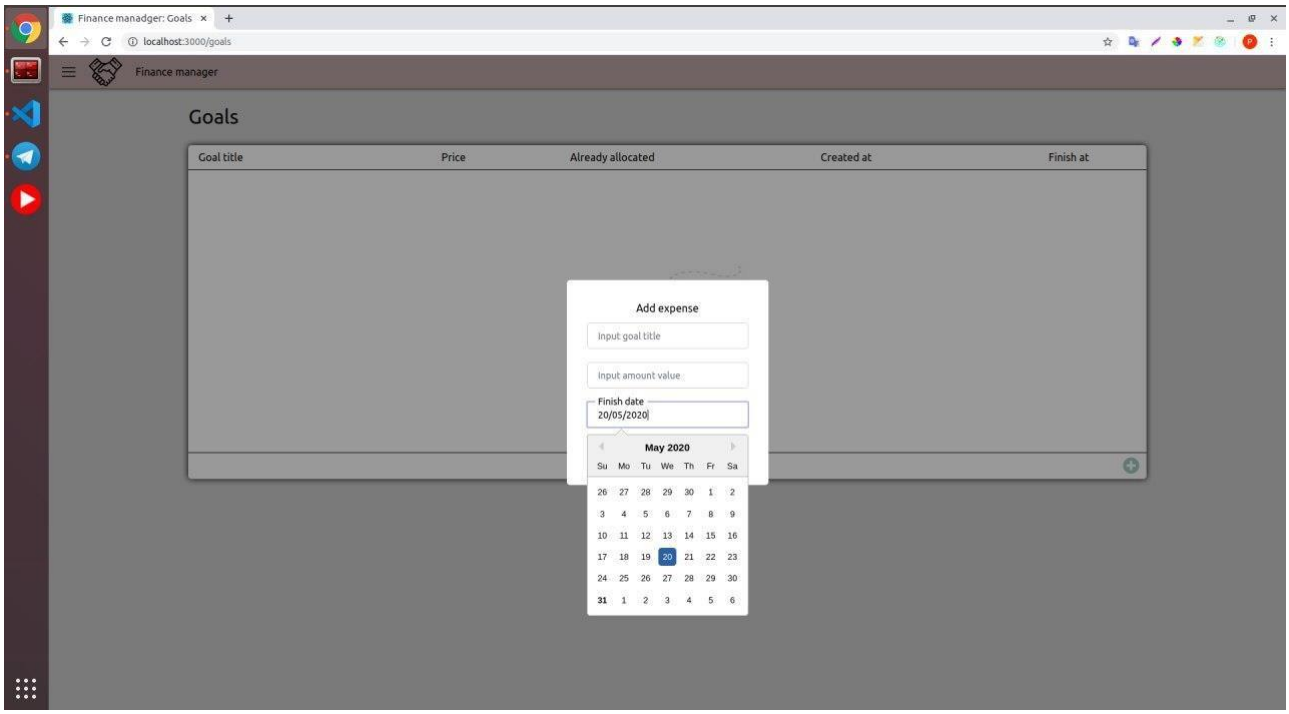


Рис. 3.15 - Інтерфейс додавання цілі

Під час створення цілі виконується наступна валідація даних на сервері:

- перевіряється дата отримання цілі на майбутній час;
- розраховується середнє значення прибутків та витрат за період існування користувача;
- сервер виконує розбиття ціни товару на обраний період;
- виконується перевірка, чи сума середнього значення витрат та місячного виділення коштів не перевищують поточне значення середнього доходу.

Під час створення одного елементу з трьох перерахованих елементів сервер проводить відповідні перевірки та повертає список разом із створеним елементом або повідомлення про помилку.

Наступною сторінкою є сторінка профілю, яка зображена на рисунку 3.16.

На ній розміщена головна інформація про користувача така як:

- ID користувача;
- електрона пошта;
- ім’я;
- загальне значення прибутків та витрат;
- значення прибутків і витрат в поточному місяці;
- графи співвідношення.

Перший граф (рисунок 3.17) відображає відношення поточного значення прибутку із врахуванням витрат та із значенням самих витрат.

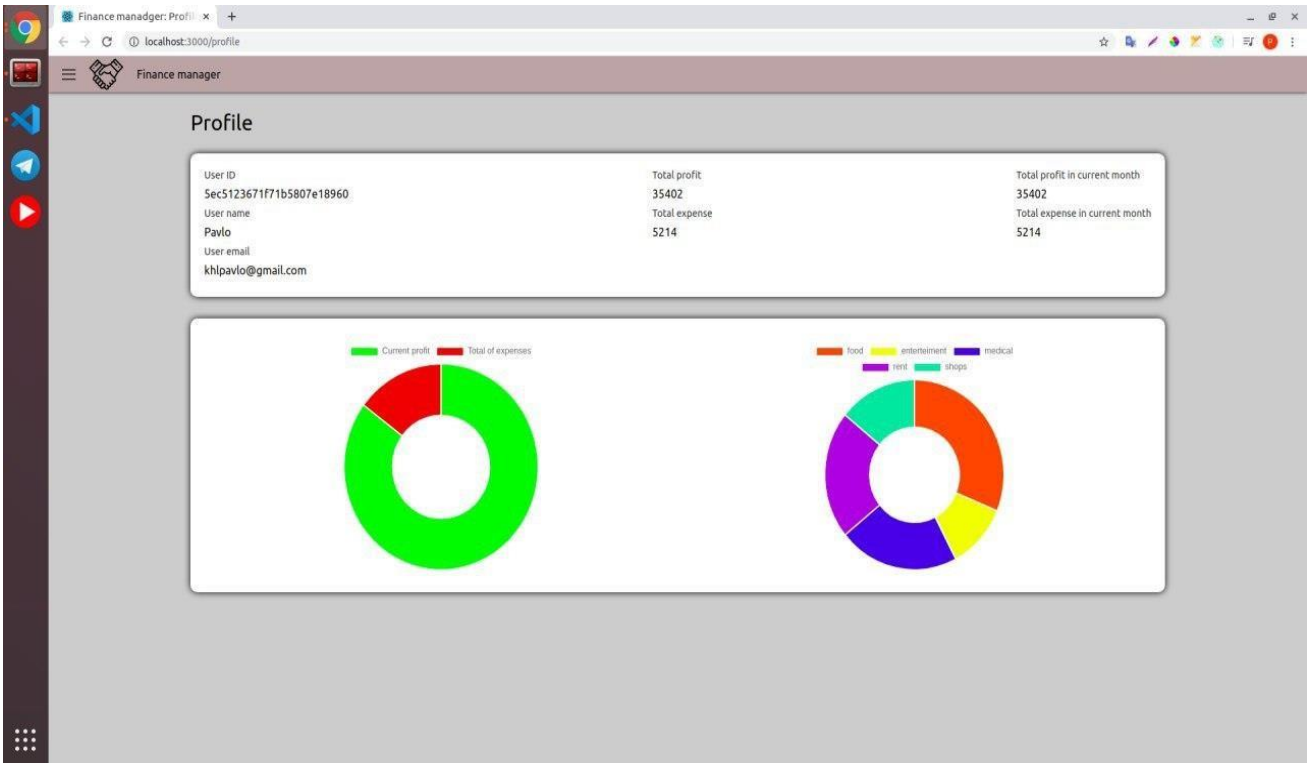


Рис. 3.16 – Інтерфейс сторінки профілю

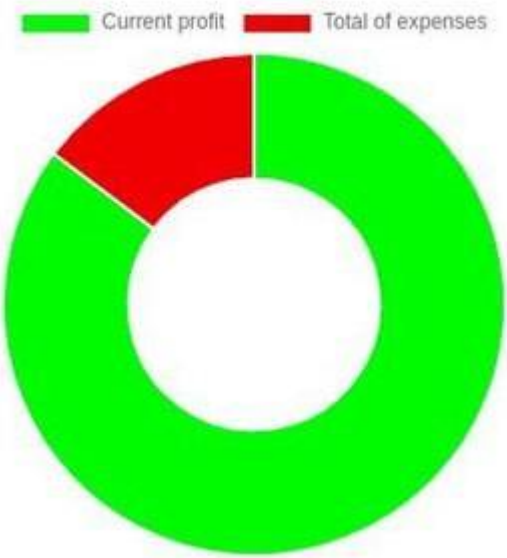


Рис. 3.17 – Граф співвідношення поточного прибутку до витрат

Другий граф відповідає співвідношенню витрат одне до одного (рисунок 3.18). При аналізі даних можна вилучити дані певної секції, для порівняння обраних типів даних.

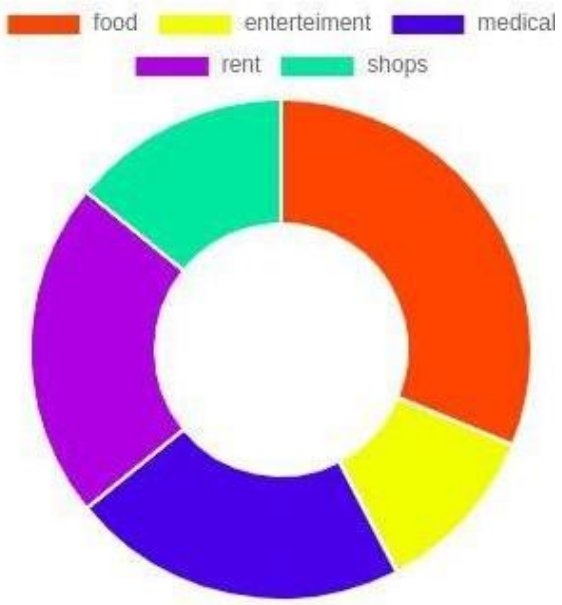


Рис. 3.18 – Граф співвідношення витрат

ВИСНОВКИ

Метою даного дипломного проєкту була розробка вебдодатку для обліку коштів та планування витрат. Розроблений вебдодаток надає змогу користувачам контролювати свої кошти при можливості враховувати майбутні придбання.

В результаті аналізу технології для розробки вебдодатків було доведено доцільність використання бібліотеки React для виконання клієнтської частини та фреймворку Express та базу даних MongoDB для серверної частини програми.

Розроблений вебдодаток дозволяє:

- реєстрацію та авторизацію користувача;
- додавання та видалення витрат;
- додавання та видалення прибутків;
- отримання графічних діаграм для аналізу обліку коштів.

Даний вебдодаток може бути реалізована в мережі Інтернет для десктопних та мобільних пристроїв.

Список використаної літератури

1. Что такое Javasctipt? MDN web docs. URL: https://developer.mozilla.org/ru/docs/Learn/JavaScript/%D0%9F%D0%B5%D1%80%D0%B2%D1%8B%D0%B5_%D1%88%D0%B0%D0%B3%D0%B8/What_is_JavaScript (дата звернення: 04.05.2020).
2. Введение в JavaScript. JavaScript.ru. URL: https://learn.javascript.ru/intro (дата звернення: 04.05.2020).
3. What is TypeScript? Typescriptlang. URL: https://www.typescriptlang.org/ (дата звеpенення: 23.05.20).
4. Основы React: всё что нужно знать для начала работы. Хабр. URL: https://habr.com/ru/company/ruvds/blog/343022/ (дата звернення: 04.05.2020).
5. Что такое Node.js и где применяется эта технология. Блог Хекслета. URL: https://ru.hexlet.io/blog/posts/zachem-izuchat-node-js-ili-o-perspektivah-bekenda-na-javascript (дата звернення: 04.05.2020).
6. Краткое руководство по Redux для начинающих. Tproger. URL: https://tproger.ru/translations/redux-for-beginners/ (дата звернення: 18.04.2020).
7. Fast, upopinionated, minimalist web framework for Node.js. ExpressJS. URL: https://expressjs.com/ (дата звернення: 04.05.2020).
8. Choosing the right Node.js Framework. Nodesourse. URL: https://nodesource.com/blog/Express-Koa-Hapi (дата звернення: 04.05.2020).
9. What is Database. Oracle. URL: https://www.oracle.com/database/what-is-database.html (дата звернення: 18.04.2020).

10. Вебсайт та його застосування. WebTec. URL: http://www.webtec.com.ua/uk/articles/index/view/2011-05-05/web-site (дата звернення: 24.05.2020)
11. What is JSON Web Token? JWT. URL: https://jwt.io/introduction/ (дата звернення: 18.05.2020).
12. The complete guide to using localStorage in JavaScript apps. LogRocket Frontend Monitoring. URL: https://blog.logrocket.com/the-complete-guide-to-using-localstorage-in-javascript-apps-ba44edb53a36/ (дата звернення: 18.05.2020).